Constrained Optimization and Radial Basis Functions in Computational Engineering

by

Graeme Schmidt

A dissertation submitted to the

Faculty of Graduate and Postdoctoral Affairs

in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

in

Aerospace Engineering

Carleton University

Ottawa, Ontario

The undersigned hereby recommends to the

Faculty of Graduate and Postdoctoral Affairs

acceptance of the dissertation

Constrained Optimization and Radial Basis Functions in Computational Engineering

submitted by **Graeme Schmidt**

in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

Professor John Nash, External Examiner

Professor John Goldak, Thesis Supervisor

Professor Tarik Kaya, Thesis Co-supervisor

Professor Ronald Miller, Chair,
Department of Mechanical and Aerospace Engineering

Department of Mechanical and Aerospace Engineering

Carleton University

May, 2019

Abstract

Duality and radial basis functions (RBFs) are compelling principles which are currently underrepresented in computational engineering. Following an introduction to the mathematics, these methodologies are applied to a series of toy problems.

To begin, both methodologies are integrated into a predictor-corrector algorithm, which is then used to model transient thermal diffusion in a one-dimensional domain. Following discretization of the heat equation with the finite element method (FEM) or RBFs, Euclidean temperature errors of 9.613E-2 (RBF) and 1.442E-1 (FEM) are observed at five seconds and 7.635E-2 (RBF) and 1.136E-1 (FEM) at ten seconds. In addition, duality is successfully applied as an error bound on the numerical solutions; the duality gap is cheaply computed by the algorithm and converges to zero as the thermal flux and temperature fields iteratively approach their exact solutions.

The RBF methodology is next applied to the interpolation of two complicated data sets. The first is a two-dimensional velocity field associated with a scattered collection of particles in a rheometer. Using a subset of these particles and their velocities as input, the RBFs accurately interpolate the scattered velocity data back on to the original collection of particles. The addition of a high-order polynomial to the RBFs is seen to improve accuracy further with an increase in algorithm complexity. The second test interpolates two-dimensional stress and stress gradient tensor fields of scattered particles on a plate in uniaxial tension. In supplement to the standard

RBF technique of the first test, a Hermite RBF technique is devised which uses given stress *and* stress gradient data as input. With respect to accuracy, the Hermite technique achieves an improvement of one order of magnitude in stress interpolation and one to two orders of magnitude in stress gradient interpolation over the standard technique.

In the final tests, RBFs are tested alongside finite differences in the discretization of two-phase heat exchange problems. In the first two cases, the RBFs more accurately compute the position of the planar phase interface with time, as well as the temperature fields in the vapour and liquid-phases. In the third case, it is found that the RBFs track the instantaneous position and velocity of a spherical bubble with higher accuracy.

For my grandmother, who taught me the value of persistence, dedication, and never giving up. Ruhe in frieden Oma, ich liebe dich.

Acknowledgments

I would like to extend my utmost gratitude to my thesis supervisors, Dr. John Goldak and Dr. Tarik Kaya. They have had an immense impact on the last years seven years of my life and, with their guidance and wisdom, I never felt that I was alone in my journey.

I would also like to thank my colleagues, Hossein Nimrouzi, Komeil Kazemi, Stanislav Tchernov, and Jianguo Zhou for their help with software troubleshooting. Thanks also to my friend and colleague Matthew Huebner, who was always there to remind me to laugh during the difficult times. Further, I owe many thanks to Carleton University and the Department of Mechanical and Aerospace Engineering for seven years of fond memories, and for the many research and teaching assistantships that they were so kind to grant me.

Finally, I would like to acknowledge the financial support provided by the Natural Sciences and Engineering Research Council of Canada discovery grants program.

Contents

List of Tables

List of Figures

Conventions and Acronyms

In the mathematics that follows, matrices will be represented with boldfaced uppercase letters, vectors with boldfaced lowercase letters, and scalars with regular typeface. The $\square^{\mathrm{T}}$ notation will be used to identify the transpose of a vector or matrix. $\mathbb{N}$ signifies the natural numbers and $\mathbb{R}$ the real numbers. With regard to intervals, such as in position or time, the convention $(\cdot, \cdot)$ or $[\cdot, \cdot]$ will be used to indicate an open or closed interval of the real numbers, respectively. For example, the position interval $x \in (0, 10]$ would be equivalent to $0 < x \leq 10$.

Set-builder notation, indicated with curled braces $\{\square\}$, will be used throughout in conjunction with the following logical symbols:

- $\in \square$, "is a member of the set/space $\square$"
- $|$, "such that"
- $\Big|_{\square}$, "evaluated at argument $\square$"
- $\square_1 \cup \square_2$, "union of the sets/spaces $\square_1$ and $\square_2$"
- $\square_1 \cap \square_2$, "intersection of the sets/spaces $\square_1$ and $\square_2$"
- $\forall \square$, "for all indices/vectors/matrices/etc. named $\square$". Alternatively, "for each index/vector/matrix/etc. named $\square$"
- $\exists \square$,"there exists a vector/matrix/set/etc. named $\square$".

Symbolic constants and variables will be defined at the beginnings of the chapters and after the first equations in which they appear. The following is a list of acronyms and initialisms used throughout:

Symbol	Definition
BC	Boundary Condition
BVP	Boundary Value Problem
FDM	Finite Difference Method
FEM	Finite Element Method
FVM	Finite Volume Method
KA	Kinematically Admissible or Kinematic Admissibility
HSSVD	Hilbert-Schmidt Singular Value Decomposition
KKT	Karush-Kuhn-Tucker
LATIN	LArge Time INcrement
LFT	Legendre-Fenchel Transform
PDE	Partial Differential Equation
QP	Quadratic Programming or Quadratic Program
PHS	PolyHarmonic Spline
RBF	Radial Basis Function
RBK	Radial Basis Kernel
ROM	Reduced Order Model
SA	Statically Admissible or Static Admissibility

SLE	System of Linear Equations
SPH	Smoothed Particle Hydrodynamics
SVD	Singular Value Decomposition

Chapter 1

Introduction

Section 1.1 opens the chapter with the engineering and mathematical background underpinning the research performed in this thesis. Section 1.2 follows with a survey of the literature which preceded and motivated the research. Section 1.3 closes the chapter with the primary objectives and novel contributions of this thesis.

1.1 Background

The topic of convex optimization is of great importance in physics and engineering. Although it may seem restrictive to limit the discussion to only convex functions, where each local minimum is a global minimum[1] of the function, it turns out that they describe the energy contained in a wide array of physical systems. For the simple example of a solid body undergoing heat conduction at steady-state, there exist unique temperature and thermal flux fields which minimize the thermal energy of the body *and* satisfy the heat equation. Stated mathematically, the *saddle point* is characterized by the set of *primal variables* (thermal fluxes) and *dual variables* (temperatures) which simultaneously minimizes a convex function (thermal energy)

[1]The problem of maximizing a concave function, where each local maximum is a global maximum of the function, also falls under the umbrella of convex optimization.

and satisfies a set of linear constraints (heat equation). The necessary conditions for identifying this saddle point are called the Karush-Kuhn-Tucker (KKT) conditions. The dual variables, alternatively called Lagrange multipliers, enforce the constraints and arise from the mathematical principle of *duality*. In combination with the constrained optimization framework introduced above, duality is a powerful yet seldom acknowledged tool in computational engineering.

Another important class of problems in computational engineering is the interpolation of scattered data. In essence, an unstructured set of M test coordinates in m-dimensional space is accompanied by a set of given data; for example, temperatures measured at M random locations in a heated slab of metal. The objective is to identify an interpolating function which matches the given data at the test coordinates. Radial Basis Functions (RBFs) have become a popular tool for scattered data interpolation, wherein a function with translational and rotational symmetry is centered at each of the test coordinates. In addition to these symmetric properties, a wide variety of basis functions exist with each having desirable traits, such as infinite-differentiability or positive definiteness. Regardless of the RBF type, they generally decrease monotonically with distance from the test coordinate about which they are centered. The interpolant is then formed from a linear combination of these M RBFs and evaluated at the desired interpolation point . To date, RBF interpolation has been used extensively in domains ranging from surface reconstruction [1][2][3] to neural networks [4].

In the 1980s, Edward Kansa [5][6] adapted the RBF methodology to the solution of Partial Differential Equations (PDEs). Just as the Finite Element Method (FEM) uses a weighted, finite set of compactly-supported test functions to approximate the solution of a PDE, RBFs may be used for the same purpose. The most significant difference between the two techniques is that the RBFs are *meshfree*; whereas the Finite

Volume Method (FVM) and FEM rely upon a mesh of discrete volumes formed from a grid of nodes, the RBF approach requires only the latter. This trait is particularly useful in Computational Fluid Dynamics (CFD) with Lagrangian reference frames, wherein the FVM and FEM meshes may become distorted leading to numerical instability. The Finite Difference Method (FDM) also struggles in such circumstances as it is best suited to structured, rectangular grids. Although it has found success in CFD, adoption of the RBF approach for the modelling of heat transport has inexplicably lagged.

1.2 Literature Survey

1.2.1 Constrained Optimization, Duality, and the LATIN Algorithm

Strang [7] discussed the mathematics of constrained optimization and its applications in physics and engineering. The concepts of Lagrange multipliers and duality were introduced with a simple, two-dimensional constrained optimization problem. This was followed by a more general mathematical treatment, wherein it was shown that the primal and dual variables are linked by a system of "equilibrium equations". Strang used these equilibrium equations as a stepping stone for proving the concepts of strong and weak duality.

In [8], Strang introduced the same fundamental concepts of optimization in a slightly different manner, starting instead with an application to projections and least squares problems. He discussed the KKT conditions, which were shown to be a generalization of the equilibrium equations explored in [7]. Given the textbook's emphasis on computational methods, Strang also reviewed a wide array of optimization

applications ranging from quantum mechanics to finite element error estimation.

Boyd and Vandenberghe [9] described in detail the properties of convex sets and functions, as well as the application of these properties to optimization problems. Perhaps most importantly, they showed that a local optimum of a convex function is also the global optimum. A comprehensive treatment of duality was given, followed by applications ranging from experiment design to entropy minimization. Solution techniques for convex optimization problems were also addressed.

Prager and Synge [10] used duality to obtain bounds on the exact solutions of elasticity problems. To begin, they defined each of the six components of the Cauchy stress tensor as functions of position throughout a given geometry (likewise for the strain tensor). Any arbitrary selection of these six stress (resp. strain) functions was collectively called a stress (resp. strain) state. The article then introduced two function spaces: 1) a space containing the stress/strain states which satisfy the displacement boundary conditions and compatibility equation and 2) a space containing the stress/strain states which satisfy the traction boundary conditions and equations of equilibrium. They showed that the exact solution state, which lies at the intersection of the two function spaces, satisfies all four of those conditions and is generally incalculable. However, it was also shown that the stress/strain states from spaces 1) and 2) provide upper and lower bounds, respectively, on the exact solution. The paper concluded with a practical application of these concepts to the torsion of a tapered prismatic bar.

Expanding upon the theme of [10], Ladevèze and Pelle [11] introduced the notion of admissible approximate solutions, wherein a set of solutions were defined which satisfy a *subset* of the governing equations and boundary conditions. For example, a Statically Admissible (SA) solution would satisfy the equilibrium equation and traction boundary conditions in a linear elastic problem, whereas Hooke's law and

the Dirichlet boundary conditions would not be satisfied. From this SA solution, a constitutive relation error arises which measures the residual in Hooke's law (or a constitutive law in general). From the constitutive relation error, Ladevèze derived a scalar bound on the error, measured in the energy norm, between the SA solution and exact solution.

In [12], Ladevèze and Perego built upon the admissible approximate solution approach. They used the LATIN (LArge Time INcrement) scheme for the discretization of space and time in elastoplastic and viscoplastic problems. The set of governing equations were partitioned into linear and non-linear equations; the stress/strain field pairs that satisfied the non-linear equations and Dirichlet boundary conditions were called Kinematically Admissible (KA) and the stress/strain field pairs that satisfied the linear equations and traction boundary conditions were called SA. Starting with a KA pair of fields, the algorithm iteratively projected between the spaces of KA and SA fields and ultimately converged to the exact discrete solution.

Maier [13] analyzed the stresses and strains in elastic and plastic structures as a quadratic programming (QP) problem (a special case of the convex programming problem class). In addition to the KA and SA states emphasized in [11] and [12], Maier also invoked the notions of convex functions and duality. Specifically, he discussed the primal problem of minimizing the potential energy and the dual problem of maximizing the complementary potential energy. He then interpreted the "true" stress response, which satisfies all of the governing equations and is thus SA and KA, as the unique state which solves the primal and dual problems.

In [14], Touchette expounded the Legendre-Fenchel Transform (LFT), a generalization of the Legendre transform commonly used in physics and thermodynamics. After defining the scalar function $f(x) : \mathbb{R} \mapsto \mathbb{R}$ with $x \in \mathbb{R}$, Touchette introduced the LFT of $f(x)$ as the scalar function $f^*(k) = \sup_{k \in \mathbb{R}}(kx - f(x))$ with $k \in \mathbb{R}$.

Note that sup(□) is the supremum used in set theory and is roughly analogous to max(□). The paper then gave a graphical demonstration of two important features of the LFT: 1) $f^*(k)$ is convex regardless of the behaviour of $f(x)$ and 2) the LFT of $f^*(k)$, symbolized by $f^{**}(x)$, is always convex and $f(x) = f^{**}(x)$ holds only if $f(x)$ is also convex. The first of these features of the LFT has significant implications for the intimately related concept of duality. Namely, given the generally non-convex primal problem of minimizing $f(x)$ over x, the dual problem of maximizing $f^*(k)$ over k is always convex. Moreover, if $f(x)$ is convex then strong duality exists between the primal and dual problems and they may be freely exchanged.

Benzi, Golub, and Liesen [15] gave a detailed account of numerical methods for the solution of QPs. They demonstrated how these QPs yield a System of Linear Equations (SLE) following application of the KKT conditions, the solution of which gives the saddle point. Solution techniques for these saddle point equations comprise direct, iterative, segregated, and coupled approaches. The segregated approach works in *either* the primal or dual variable space, while the coupled approach *simultaneously* solves the SLE for the primal and dual variables. In addition, three practical problems (namely incompressible fluid flow, least squares approximation, and interior point methods) from the domain of computational engineering were introduced and formulated as saddle point problems.

Auricchio, Brezzi, and Lovadina [16] presented a mixed FEM approach for the solution of elliptic PDEs. They started by introducing the potential energy functional with either two or three arguments, respectively called the Hellinger-Reissner and Hu-Washizu formulations. To enforce stationarity of these two potential energy functionals, the calculus of variations was applied and the resulting functions were equated with zero. The resulting SLE was shown to be a saddle point problem, the

solution of which minimized the system's potential energy and satisfied the governing PDE. Furthermore, this methodology was demonstrated in three different elliptic PDEs: thermal diffusion, Stokes flow in fluid mechanics, and linear elasticity in solid mechanics. In a broader sense, [15] and [16] provided an alternate route to the saddle point SLE with respect to the duality approach advocated by Strang in [7] and [8].

1.2.2 RBF Scattered Data Interpolation and Discretization of PDEs

Fornberg and Flyer [17] gave a comprehensive account of RBFs, prefaced by a brief summary of finite difference and pseudospectral methods for discretization of PDEs. RBFs were then introduced as a generalization of these two methods, with the compelling advantage of not requiring simple, structured grids of nodes. They then distinguished between globally and locally-supported RBFs, with the former being non-zero on all M nodes in the grid and the latter non-zero only on $k < M$ of those nodes and zero elsewhere. The local-support for a desired interpolation point $\mathbf{r}$, existing in the same m-dimensional space as the nodes, consists of the k nodes nearest to it and is called the k-stencil of $\mathbf{r}$. Both of these RBF types were then used in the discretization and solution of PDEs defined on spheres, where they performed favourably compared with the finite difference and pseudospectral methods.

Fasshauer and McCourt's textbook [18] gave a detailed analysis of the mathematical theory of RBFs. The highlights of this analysis were as follows:

- The "uncertainty principle" for infinitely-differentiable (symbolically, C^∞) RBFs, where small shape factors (labelled ϵ) are found to give more accurate interpolants but the SLE for the RBF weight factors becomes ill-conditioned. Conversely, large shape factors are found to give a well-conditioned SLE but

the interpolant becomes less accurate.

- When the C^∞ RBFs are decomposed with the Hilbert-Schmidt Singular Value Decomposition (HSSVD), the interpolant may be expressed as an equivalent linear combination of the eigenfunctions of the RBF; Fasshauer called this a "stable basis". With the stable basis, the ill-conditioning associated with the small shape factors may be mitigated.

- Infinitely-differentiable RBFs may be interpreted as generalizations of polynomials and finitely-differentiable RBFs may be interpreted as generalizations of piecewise polynomial splines.

- With the Hermite interpolation approach, the interpolant is formed from a linear combination of the RBFs themselves (as in the standard approach outlined on page 5) and a linear combination of RBF derivatives of order $1, ..., d$. This Hermite approach gives rise to both symmetric and asymmetric (Kansa's method) SLEs for the RBF and RBF derivative weight factors.

[18] complemented the abstract points above with a wide variety of practical applications of RBFs, ranging from surrogate modeling to boundary/initial value problems and support vector machines.

Fornberg et al. [19][20][21] and Cavoretto et al. [22] each addressed the issue of selecting shape factors for the infinitely-differentiable class of RBFs. More precisely, they proposed techniques to mitigate the uncertainty principle introduced above for "flat" RBFs (i.e. for $\epsilon \to 0$). These techniques each replace the standard RBFs ϕ with a set of stable basis functions ψ which span the same function space $\mathscr{F}$; expressed mathematically, $\mathscr{F} = \text{span}[\phi(x, x_i)]$ is replaced with $\mathscr{F} = \text{span}[\psi(x, x_i)]$ at each node in the grid $x_{i=1,...,M}$. [19], [20], and [21] advocated that ψ be formed

from the QR decomposition of the Gaussian RBFs and [22] proposed that ψ be formed from the HSSVD of the "iterated Brownian bridge kernels" which were defined within that article. Regardless, all four papers found that interpolation with these stable basis functions coincides with polynomial interpolation (or spherical harmonics interpolation when the nodes and data were scattered over a sphere) in the limit $\epsilon \to 0$.

Flyer et al. appended a polynomial of degree d to the linear combination of locally-supported RBFs for scattered data interpolation [23] and solution of PDEs [24]. For interpolation with stencils of size k, they suggested that d be chosen as high as possible with the constraint that the number of terms in the polynomial be $\leq k-1$ to ensure uniqueness of the polynomial. Contrarily, for the purpose of PDE discretization, the authors recommended a polynomial with no more than $(k-1)/2$ terms. Using this methodology, [23] interpolated a simple test function using PolyHarmonic Spline (PHS) and Gaussian type RBFs both with polynomials of varying degrees. They found that both RBF types achieved high accuracy but preferred the PHS type with a maximal-degree polynomial because, in contrast with the Gaussian type, they do not require selection of a shape factor. [24] concluded with a benchmark test from the field of geophysics, using PHS RBFs and polynomials with variable degree for spatial discretization. The results validated their assumption that approximately $(k-1)/2$ polynomial terms yield the best accuracy and demonstrated that even modestly sized stencils of $k = 19$ and 37 (from a total of $M = 40{,}401$ test coordinates) give good accuracy in two dimensions.

In Stevens et al. [25][26][27] and Wright and Fornberg [28], an interpolating function was constructed from the standard linear combination of RBFs and a linear combination of RBF derivatives. Called Hermite interpolation, this approach was adapted to Boundary Value Problems (BVPs) by reinterpreting the interpolant as an unknown field $u(\mathbf{r})$ on an m-dimensional domain with a closed boundary. The domain

was discretized with a grid of $M + M'$ test coordinates: the RBFs were centered at test coordinates $\mathbf{r} = \boldsymbol{\alpha}_{i=1,\ldots,M}$ and the RBF derivatives were centered at test coordinates $\mathbf{r} = \boldsymbol{\beta}_{j=1,\ldots,M'}$. Likewise, the boundary was discretized with test coordinates wherein the given boundary conditions were enforced. For an arbitrary location $\mathbf{r}$ in the domain, the locally-supported stencils were modified to include the k RBFs and k' RBF derivatives nearest to $\mathbf{r}$. The following steps were then taken:

- At an arbitrary test coordinate $\boldsymbol{\alpha}_i$, the linear combination of k nearest RBFs and k' nearest RBF derivatives was equated with $u(\mathbf{r} = \boldsymbol{\alpha}_i)$.
- At an arbitrary derivative test coordinate $\boldsymbol{\beta}_j$, the continuous differential operator specified by the BVP (e.g. the Laplacian in Poisson's equation) was applied to the linear combination of k nearest RBFs and the linear combination of k' nearest RBF derivatives. The result was equated with the given right-hand side of the BVP (e.g. the source/sink term in Poisson's equation) evaluated at $\boldsymbol{\beta}_j$.

Iteration of step one at $\boldsymbol{\alpha}_{i=1,\ldots,M}$ and step two at $\boldsymbol{\beta}_{j=1,\ldots,M'}$, collectively referred to as *collocation*, resulted in a SLE which could then be solved for $u(\mathbf{r} = \boldsymbol{\alpha}_{i=1,\ldots,M})$. This methodology was applied in [25], [26], and [27] to solve BVPs concerning mass advection-diffusion-reaction and the infiltration of water into porous media, while [28] solved Poisson's equation on a variety of two-dimensional geometries.

1.3 Research Objectives

Sections 1.1 and 1.2 allowed for the identification of three novel topics of research, the first concerning the LATIN algorithm pioneered by Pierre Ladevèze. In [12], the "duality preserving" nature of LATIN was briefly explored. However, Ladevèze's

discussion tended to conflate this property with a very technical treatment of elasto-plastic and viscoplastic constitutive relationships, which made it difficult to decipher his intent. Giulio Maier [13] addressed convexity and duality in more detail than [12], but his discussion of these topics was also obscured at times by excessive technical details. The first objective of this thesis is to build upon [12] and [13] in two ways: 1) rework the LATIN algorithm as a discrete convex optimization problem in a way that is intuitive, yet mathematically detailed 2) construct an error bound derived *explicitly* from the principle of duality and thereby demonstrate its value in computational engineering.

The second objective of this thesis is to apply RBFs to the interpolation of scattered data. The breadth of existing research on this topic leaves little room for novel applications, so emphasis will instead be placed on RBF interpolation of more complex data fields than are typically found in the literature. Specifically, a velocity vector field will be interpolated in Section 5.1 and stress and stress gradient tensor fields will be interpolated in Section 5.2. In the latter case, the inputs to the problem will include a discrete stress field and a discrete stress gradient field, both defined on a primary grid of nodes. Two techniques will be used to interpolate the given data sets to a secondary grid: 1) the standard RBF technique which uses only the primary stress field as input and 2) the Hermite RBF technique which uses both the primary stress and stress gradient fields as inputs. The resultant stress and stress gradient fields on the secondary grid will be compared with a reference solution to assess the accuracy of the standard and Hermite techniques.

The final objective of this thesis is to demonstrate RBF discretization of PDEs, in particular two-phase heat exchange. In this class of problems, an interface separates the two phases (vapour and liquid for example) and translates as the vapour condenses or liquid vapourizes. Depending upon the problem, the temperature in one phase is

held constant and the other phase is governed by the advection and diffusion of heat. In a Lagrangian or semi-Lagrangian formulation, the grid of nodes is advected by bulk fluid motion, making RBFs an attractive means for the discretization of space. This is because RBFs are meshless and amenable to unstructured node placement; the same respective traits that FEM/FVM and FDM lack, thus making modelling of advection difficult. In Chapter 6, FDM and RBFs will be used to discretize three two-phase heat exchange problems of incremental complexity, with the goal of comparing the accuracy and robustness of the two methods.

In Chapter 4, a simple, one-dimensional, transient thermal diffusion problem will be used to unify these three objectives. Specifically, this toy problem will demonstrate that the constrained optimization, duality, and RBF methodologies may be easily combined within the LATIN algorithm to accurately solve continuum mechanics problems. To this end, the LATIN numerical solution will be compared with a semi-analytical solution in order to substantiate its viability.

In light of the research objectives described above, this thesis might be summarized as follows: *convex optimization, duality, and RBFs are combined or compared with well-established methods, such as FDM and FEM, in the solution of toy problems.* The results of these test problems will be assessed primarily by comparison with available analytical or semi-analytical solutions. Other performance characteristics such as error convergence rate and algorithm computational complexity will not be formally pursued.

Chapter 2

Constrained Optimization, Duality, and the LATIN Algorithm

Chapter 2 begins with a discussion of constrained optimization in the primal and dual spaces. The optimization of convex functions is then introduced as a simple yet important class of problems in computational engineering. The LATIN algorithm is then explored as a practical application of convex optimization and duality. Namely, the duality concept is used to derive an error bound for the LATIN algorithm.

General symbols:

Symbol	**Definition and (Units)**
$\square_1 \otimes \square_2$	tensor product of vectors or matrices $\square_1$ and $\square_2$
$\|\| \cdot \|\|_F$	Frobenius matrix norm
$\mathbf{0}^m$	column vector of zeros of length m
$\mathbf{0}^{mxn}$	matrix of zeros of size m x n
$\mathbf{A} \in \mathbb{R}^{mxn}$	discrete gradient operator (m^{-1})

$c_p \in \mathbb{R}^+$	isotropic specific isobaric heat capacity of Ω ($\mathrm{Jkg^{-1}K^{-1}}$)
$\mathbf{C}^{-1} \in \mathbb{R}^{mxm}$	inverse thermal conductivity tensor ($\mathrm{W^{-1}mK}$)
$d \in \mathbb{R}^+$	duality gap
$\widetilde{d} \in \mathbb{R}^+$	modified duality gap
$d\mathbf{a} = \mathbf{n} \cdot da$	infinitesimal area da with outward-oriented normal vector $\mathbf{n}$ ($\mathrm{m^2}$)
dV	infinitesimal volume ($\mathrm{m^3}$)
$\mathrm{div}(\square)$	divergence operator on the field or vector $\square$
$D(\boldsymbol{\lambda}, \boldsymbol{\mu}) :$ $\mathbb{R}^n$ x $\mathbb{R}^p \mapsto \mathbb{R}$	scalar dual function
$D(\mathbf{U}) : \mathscr{U}_h \mapsto \mathbb{R}$	dual function which maps a temperature field in discrete spacetime to a scalar energy (J)
$\mathbf{D} \in \mathbb{R}^{nx\tau}$	LATIN temperature correction matrix (K)
$\widetilde{\mathbf{D}} \in \mathbb{R}^{Kx\tau}$	ROM of $\mathbf{D}$ (K)
$f_j(\mathbf{x}) : \mathbb{R}^m \mapsto \mathbb{R}$	j^{th} scalar equality constraint function
$\mathbf{F}_1 \in \mathbb{R}^{mx\tau}$	matrix representing the Dirichlet BCs ($\mathrm{Km^{-1}}$)
$\mathbf{F}_2 \in \mathbb{R}^{nx\tau}$	matrix representing the sensible heat rate, volumetric source/sink magnitudes, and Neumann BCs (J)
$g_k(\mathbf{x}) : \mathbb{R}^m \mapsto \mathbb{R}$	k^{th} scalar inequality constraint function
$\mathrm{grad}(\square)$	gradient operator on the field or vector $\square$
$\mathbf{G}$	Gramian matrix

$\mathbf{h}_i \in \mathbb{R}^n$	i^{th} left singular vector of the matrix $\mathbf{R}_2$
$\mathbf{H} \in \mathbb{R}^{n\text{x}n}$	matrix with the left singular vectors $\mathbf{h}_{i=1,\dots,n}$ as its columns
$\widetilde{\mathbf{H}} \in \mathbb{R}^{n\text{x}K}$	ROM of $\mathbf{H}$
$\mathbf{H}(\square) \in \mathbb{R}^{m\text{x}m}$	Hessian matrix of the function $\square$[2]
$\mathbf{I}^m$	identity matrix of size m x m
im($\square$)	image of the matrix $\square$
$\mathscr{K}_D \subset \mathbb{R}^{n\text{x}\tau}$	manifold of all dual temperature fields, in discrete spacetime, which satisfy the Dirichlet BCs
$\mathscr{K}_P \subset \mathbb{R}^{m\text{x}\tau}$	manifold of all primal flux fields, in discrete spacetime, which satisfy Fourier's law
$K \in \mathbb{N}$	number of singular values and left/right singular vectors used in the ROM of $\mathbf{R}_2$
ker($\square$)	kernel of the matrix $\square$
$\mathbf{K} \in \mathbb{R}^{n\text{x}n}$	discrete diffusion operator (JK^{-1})
$\widetilde{\mathbf{K}} \in \mathbb{R}^{K\text{x}K}$	ROM of $\mathbf{K}$ (JK^{-1})
l	length of the one-dimensional domain (m)
$\mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\mu})$: $\mathbb{R}^m$ x $\mathbb{R}^n$ x $\mathbb{R}^p \mapsto \mathbb{R}$	scalar Lagrangian function

[2]Note the abuse of notation between the Hessian matrix, with its argument ($\square$), and the left singular matrix which has no argument. In the chapters that follow, the meaning of $\mathbf{H}$ will be made clear by the context in which it is used.

$\mathcal{L}[\mathbf{q}, u]$: $\mathscr{Q}$ x $\mathscr{U} \mapsto \mathbb{R}$	Lagrangian functional which maps a flux/temperature pair in continuous spacetime to a scalar energy (J)
$\mathcal{L}(\mathbf{Q}, \mathbf{U})$: $\mathscr{Q}_h$ x $\mathscr{U}_h \mapsto \mathbb{R}$	Lagrangian function which maps a flux/temperature pair in discrete spacetime to a scalar energy (J)
$m \in \mathbb{N}$	quantity of primal variables **or** quantity of free primal nodes in Ω and on $\Gamma_{\mathfrak{D}}$
$\widetilde{m} \in \mathbb{N}$	quantity of primal nodes on $\Gamma_{\mathfrak{N}}$ with prescribed Neumann BCs
$M = m + \widetilde{m}$	total quantity of nodes in the primal grid
$\mathbf{M} \in \mathbb{R}^{nxn}$	mass matrix (JK^{-1})
$n \in \mathbb{N}$	quantity of equality constraints **or** quantity of free dual nodes in Ω and on $\Gamma_{\mathfrak{N}}$
$\widetilde{n} \in \mathbb{N}$	quantity of dual nodes on $\Gamma_{\mathfrak{D}}$ with prescribed Dirichlet BCs
$N = n + \widetilde{n}$	total quantity of nodes in the dual grid
$p \in \mathbb{N}$	quantity of inequality constraints **or** iteration number of the LATIN algorithm
$P(\mathbf{x}) : \mathbb{R}^m \mapsto \mathbb{R}$	scalar primal function
$P[\mathbf{q}] : \mathscr{Q} \mapsto \mathbb{R}$	primal functional which maps a flux field in continuous spacetime to a scalar energy (J)
$P(\mathbf{Q}) : \mathscr{Q}_h \mapsto \mathbb{R}$	primal function which maps a flux field in discrete spacetime to a scalar energy (J)

$\mathscr{Q}$	function space containing all flux fields which are continuous in spacetime
$\mathscr{Q}_h \in \mathbb{R}^{m\text{x}\tau}$	space containing all flux fields which are discrete in spacetime
$\mathbf{q}(\mathbf{r},t) \in \mathscr{Q}$	arbitrary flux field in continuous spacetime (Wm^{-2})
$\mathbf{q}^*(\mathbf{r},t) \in \mathscr{Q}$	exact flux field in continuous spacetime (Wm^{-2})
$\widetilde{\mathbf{q}}(\mathbf{r},t)$	prescribed Neumann flux field in continuous spacetime (Wm^{-2})
$\mathbf{q}_{i,k}$	flux at the i^{th} primal node and k^{th} time-step (Wm^{-2})
$\mathbf{Q} \in \mathscr{Q}_h$	arbitrary flux field in discrete spacetime (Wm^{-2})
$\mathbf{Q}^* \in \mathscr{Q}_h$	exact flux field in discrete spacetime (Wm^{-2})
$\widetilde{\mathbf{Q}} \in \mathbb{R}^{\widetilde{m}\text{x}\tau}$	prescribed Neumann flux field in discrete spacetime (Wm^{-2})
$\widehat{\mathbf{Q}} \in \mathscr{K}_P$	KA flux field in discrete spacetime (Wm^{-2})
$\overline{\mathbf{Q}} \in \mathscr{S}_P$	SA flux field in discrete spacetime (Wm^{-2})
$r \in \mathbb{N}$	rank of the matrix $\mathbf{R}_2$
$\mathbf{r} \in \Omega$	position vector (m)
$\mathbf{R}_1 \in \mathbb{R}^{m\text{x}\tau}$	Fourier's law residuals in discrete spacetime (Km^{-1})
$\mathbf{R}_2 \in \mathbb{R}^{n\text{x}\tau}$	heat equation residuals in discrete spacetime (J)
$\widetilde{\mathbf{R}}_2 \in \mathbb{R}^{n\text{x}\tau}$	ROM of the residual matrix $\mathbf{R}_2$ (J)
$\mathscr{S}_D \subset \mathbb{R}^{n\text{x}\tau}$	manifold of all dual temperature fields, in discrete spacetime, which satsify the heat equation

$\mathscr{S}_P \subset \mathbb{R}^{m\text{x}\tau}$	manifold of all primal flux fields, in discrete spacetime, which satsify the Neumann BCs
$S(\mathbf{r}, t)$	volumetric source/sink field in continuous spacetime (Wm^{-3})
$\mathbf{S} \in \mathbb{R}^{n\text{x}\tau}$	volumetric source/sink field in discrete spacetime (Wm^{-3})
$t \in \mathbb{R}^+$	time (s)
$\text{tr}(\square)$	trace of the matrix $\square$
$\mathscr{U}$	function space containing all temperature fields which are continuous in spacetime
$\mathscr{U}_h \in \mathbb{R}^{n\text{x}\tau}$	space containing all temperature fields which are discrete in spacetime
$u(\mathbf{r}, t) \in \mathscr{U}$	arbitrary temperature field in continuous spacetime (K)
$u^*(\mathbf{r}, t) \in \mathscr{U}$	exact temperature field in continuous spacetime (K)
$\widetilde{u}(\mathbf{r}, t)$	prescribed Dirichlet temperature field in continuous spacetime (K)
$u_0(\mathbf{r})$	prescribed temperature field in continuous space at $t = 0$ s (K)
$u_{i,k}$	temperature at the i^{th} dual node and k^{th} time-step (K)
$\mathbf{U} \in \mathscr{U}_h$	arbitrary temperature field in discrete spacetime (K)
$\mathbf{U}^* \in \mathscr{U}_h$	exact temperature field in discrete spacetime (K)
$\widetilde{\mathbf{U}} \in \mathbb{R}^{\widetilde{n}\text{x}\tau}$	prescribed Dirichlet temperature field in discrete spacetime (K)

$\widehat{\mathbf{U}} \in \mathscr{K}_D$	KA temperature field in discrete spacetime (K)
$\overline{\mathbf{U}} \in \mathscr{S}_P$	SA temperature field in discrete spacetime (K)
$\mathbf{U}^\dagger \in \mathscr{U}_h$	temperature lag matrix (K)
$\mathbf{v}_k \in \mathbb{R}^\tau$	k^{th} right singular vector of the matrix $\mathbf{R}_2$
$\mathbf{V} \in \mathbb{R}^{\tau \mathrm{x} \tau}$	matrix with the right singular vectors $\mathbf{v}_{k=1,\ldots,\tau}$ as its columns
$\widetilde{\mathbf{V}} \in \mathbb{R}^{\tau \mathrm{x} K}$	ROM of $\mathbf{V}$
$x \in \mathbb{R}^+$	position (m)
x_i	i^{th} primal variable
$\mathbf{x} \in \mathbb{R}^m$	arbitrary vector of primal variables
$\mathbf{x}^* \in \mathbb{R}^m$	vector of primal variables which minimizes the primal function
$\widetilde{\mathbf{x}} \in \mathbb{R}^m$	vector of perturbed primal variables

Greek symbols:

Symbol	**Definition and (Units)**
$\delta^p_\square \in \mathbb{R}^+$	$\square = P$: $P(\overline{\mathbf{Q}}^p) - \mathcal{L}(\mathbf{Q}^*, \mathbf{U}^*)$ at the p^{th} LATIN iteration (J) $\square = D$: $\mathcal{L}(\mathbf{Q}^*, \mathbf{U}^*) - D(\widehat{\mathbf{U}}^p)$ at the p^{th} LATIN iteration (J)
$\Delta R \in \mathbb{R}$	difference between $\mathbf{R}_2$ and $\widetilde{\mathbf{R}}_2$ in the Frobenius matrix norm (J)
$\Delta t \in \mathbb{R}^+$	time-step size (s)
$\boldsymbol{\epsilon}_\square$	$\square = x$: perturbation vector of length m applied to $\mathbf{x}^*$ $\square = \lambda$: perturbation vector of length n applied to $\boldsymbol{\lambda}^*$

	$\square = \mu$: perturbation vector of length p applied to $\boldsymbol{\mu}^*$
$\Gamma = \partial\Omega$	closed boundary of Ω
$\Gamma_D \subseteq \Gamma$	sub-boundary where the Dirichlet BCs are prescribed
$\Gamma_N \subseteq \Gamma$	sub-boundary where the Neumann BCs are prescribed
$\kappa \in \mathbb{R}^+$	isotropic thermal conductivity of Ω ($\mathrm{Wm^{-1}K^{-1}}$)
λ_j	KKT multiplier for equality constraint $f_j(\mathbf{x})$
$\boldsymbol{\lambda} \in \mathbb{R}^n$	arbitrary vector of equality constraint multipliers
$\boldsymbol{\lambda}^* \in \mathbb{R}^n$	vector of equality constraint multipliers which maximizes the dual function
$\widetilde{\boldsymbol{\lambda}} \in \mathbb{R}^n$	vector of perturbed equality constraint multipliers
μ_k	KKT multiplier for inequality constraint $g_k(\mathbf{x})$
$\boldsymbol{\mu} \in \mathbb{R}^p$	arbitrary vector of inequality constraint multipliers
$\boldsymbol{\mu}^* \in \mathbb{R}^p$	vector of inequality constraint multipliers which maximizes the dual function
$\widetilde{\boldsymbol{\mu}} \in \mathbb{R}^p$	vector of perturbed inequality constraint multipliers
$\nabla_{x,\lambda,\mu}$	vector operator with entries $\left[\frac{\partial}{\partial x_i} \quad \frac{\partial}{\partial \lambda_j} \quad \frac{\partial}{\partial \mu_k}\right]$, where $i = 1, ..., m$, $j = 1, ..., n,\ k = 1, ..., p$
$\Omega \subseteq \mathbb{R}^3$	three-dimensional domain
$\phi_i^{\square}(\mathbf{r})$	$\square = q$: basis function for the space component of $\mathscr{Q}_h$ $\square = u$: basis function for the space component of $\mathscr{U}_h$

	uses a local coordinate system with its origin at $\mathbf{r}_i$ in space
$\boldsymbol{\phi}^{\square}$	$\square = q$: row vector of length m containing $\phi^q_{i=1,\ldots,m}(\mathbf{r})$ $\square = u$: row vector of length n containing $\phi^u_{i=\widetilde{n}+1,\ldots,N}(\mathbf{r})$
$\widetilde{\boldsymbol{\phi}}^{\square}$	$\square = q$: row vector of length $\widetilde{m}$ containing $\phi^q_{i=m+1,\ldots,M}(\mathbf{r})$ $\square = u$: row vector of length $\widetilde{n}$ containing $\phi^u_{i=1,\ldots,\widetilde{n}}(\mathbf{r})$
$\psi^{\square}_{i,k}(\mathbf{r},t)$	$\square = q$: basis function for $\mathscr{Q}_h$ $\square = u$: basis function for $\mathscr{U}_h$ uses a local coordinate system with its origin at $\mathbf{r}_i$ in space and t_k in time
$\rho \in \mathbb{R}^+$	isotropic density of Ω (kgm^{-3})
$\sigma_k \in \mathbb{R}$	k^{th} singular value of the matrix $\mathbf{R}_2$
$\boldsymbol{\Sigma} \in \mathbb{R}^{n\mathrm{x}\tau}$	matrix with the singular values $\sigma_{k=1,\ldots,r}$ on its main diagonal and zeros elsewhere
$\widetilde{\boldsymbol{\Sigma}} \in \mathbb{R}^{K\mathrm{x}K}$	ROM of $\boldsymbol{\Sigma}$
$\tau \in \mathbb{N}$	quantity of time-steps
$\theta_k(t)$	basis function for the time component of $\mathscr{Q}_h$ and $\mathscr{U}_h$. Uses a local coordinate system with its origin at t_k in time
$\boldsymbol{\theta} \in \mathbb{R}^{\tau+1}$	column vector of length $\tau + 1$ containing $\theta_{k=0,\ldots,\tau}(t)$

2.1 Constrained Optimization

Begin by defining the vector of primal variables $\mathbf{x} = \begin{bmatrix} x_1 & \cdots & x_m \end{bmatrix}^{\mathrm{T}} \in \mathbb{R}^m$. A multivariable primal function may be defined which maps $\mathbf{x}$ to the real number line, $P(\mathbf{x}) : \mathbb{R}^m \mapsto \mathbb{R}$. The primal optimization problem may be formulated like so:

Problem P.1: given a set of equality and inequality constraints on $\mathbf{x}$*, minimize* $P(\mathbf{x})$ *over the set of all feasible* $\mathbf{x}$*.*

which may be summarized mathematically as follows

$$\begin{aligned} &\min_{\mathbf{x}} P(\mathbf{x}) \\ &\text{subject to } f_j(\mathbf{x}) = 0, j = 1, ..., n \\ &\qquad\qquad g_k(\mathbf{x}) \leq 0, k = 1, ..., p \end{aligned} \tag{P.1}$$

where $f_j(\mathbf{x}) : \mathbb{R}^m \mapsto \mathbb{R}$ are the set of equality constraint functions and $g_k(\mathbf{x}) : \mathbb{R}^m \mapsto \mathbb{R}$ are the set of inequality constraint functions. The primal and constraint functions may be assembled into a single scalar function called the Lagrangian, $\mathcal{L} : \mathbb{R}^m \text{ x } \mathbb{R}^n \text{ x } \mathbb{R}^p \mapsto \mathbb{R}$

$$\mathcal{L}(\mathbf{x}, \lambda_j, \mu_k) = P(\mathbf{x}) + \sum_{j=1}^{n} \lambda_j f_j(\mathbf{x}) + \sum_{k=1}^{p} \mu_k g_k(\mathbf{x}) \tag{1}$$

where $\lambda_j \in \mathbb{R}$ and $\mu_k \in \mathbb{R}^+$ are the KKT multipliers[3] with one multiplier for each constraint. The vectors $\boldsymbol{\lambda} \in \mathbb{R}^n$ and $\boldsymbol{\mu} \in (\mathbb{R}^+)^p$ contain the respective multipliers.

The KKT conditions [8][9] provide the necessary conditions for identifying the saddle point of the Lagrangian. Assuming P and each f_j and g_k to be continuously differentiable, the KKT conditions may be summarized mathematically as follows

[3]The λ_j are commonly referred to as Lagrange multipliers

$$\begin{aligned}
&\nabla_{x,\lambda,\mu}\mathcal{L}(\mathbf{x}^*,\boldsymbol{\lambda}^*,\boldsymbol{\mu}^*)=0\\
&f_j(\mathbf{x}^*)=0,\ \forall j\\
&g_k(\mathbf{x}^*)\le 0,\ \forall k\\
&\mu_k\ge 0,\ \forall k\\
&\mu_k g_k(\mathbf{x}^*)=0,\ \forall k
\end{aligned} \tag{2}$$

The first condition of (2) imposes stationarity of the Lagrangian at the saddle point, $(\mathbf{x}^*,\boldsymbol{\lambda}^*,\boldsymbol{\mu}^*)$. This stationarity condition may alternatively be expressed as a SLE of size $m+n+p$

$$\begin{bmatrix}\partial\mathcal{L}/\partial x_{i=1,\dots,m}\\ \partial\mathcal{L}/\partial\lambda_{j=1,\dots,n}\\ \partial\mathcal{L}/\partial\mu_{k=1,\dots,p}\end{bmatrix}=\begin{bmatrix}\mathbf{0}^m\\ \mathbf{0}^n\\ \mathbf{0}^p\end{bmatrix} \tag{3}$$

where $(\mathbf{x}^*,\boldsymbol{\lambda}^*,\boldsymbol{\mu}^*)$ is extracted by solution of the system above. Moreover, two *approximately equivalent* optimization problems arise from SLE (3):

Problem L.1: *maximize the Lagrangian over the set of all* $\boldsymbol{\lambda}$ *and positive* $\boldsymbol{\mu}$*, followed by minimization over the set of all* $\mathbf{x}$*.*

Problem L.2: *minimize the Lagrangian over the set of all* $\mathbf{x}$*, followed by maximization over the set of all* $\boldsymbol{\lambda}$ *and positive* $\boldsymbol{\mu}$*.*

which, in turn, may be summarized mathematically like so

$$\underset{\mathbf{x}}{\min}\left(\underset{\boldsymbol{\lambda},\boldsymbol{\mu}}{\max}\ \mathcal{L}(\mathbf{x},\boldsymbol{\lambda},\boldsymbol{\mu})\right)\quad\overset{?}{=}\quad\underset{\boldsymbol{\lambda},\boldsymbol{\mu}}{\max}\left(\underset{\mathbf{x}}{\min}\ \mathcal{L}(\mathbf{x},\boldsymbol{\lambda},\boldsymbol{\mu})\right) \tag{4}$$

L.1 L.2

Note that the $\overset{?}{=}$ symbol used above indicates that the conditions necessary for equality to hold will be specified later. Focusing for the moment on Problem L.1,

$\max_{\boldsymbol{\lambda},\boldsymbol{\mu}} \mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\mu})$ is governed by rows two and three of SLE (3)

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial \lambda_j} &= f_j(\mathbf{x}) = 0, \ j = 1, ..., n \\ \frac{\partial \mathcal{L}}{\partial \mu_k} &= g_k(\mathbf{x}) = 0, \ k = 1, ..., p \end{aligned} \tag{5}$$

Substitution of this result into Equation (1) simply returns the primal function, such that $\max_{\boldsymbol{\lambda},\boldsymbol{\mu}} \mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = P(\mathbf{x})$. Consequently, Problems P.1 and L.1 are identical.

Focusing on Problem L.2, $\min_{\mathbf{x}} \mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\mu})$ is governed by the first row of (3)

$$\frac{\partial \mathcal{L}}{\partial x_i} = \frac{dP}{dx_i} + \sum_{j=1}^{n} \lambda_j \frac{df_j}{dx_i} + \sum_{k=1}^{p} \mu_k \frac{dg_k}{dx_i} = 0, \ i = 1, ..., m \tag{6}$$

Solution of SLE (6) gives the minimal $\mathbf{x}^*$ vector whose entries are, in general, each functions of $\lambda_{j=1,...,n}$ and $\mu_{k=1,...,p}$. Substitution of this $\mathbf{x}^*$ into Equation (1) returns the dual function $D(\boldsymbol{\lambda}, \boldsymbol{\mu}) : \mathbb{R}^n \text{ x } \mathbb{R}^p \mapsto \mathbb{R}$, such that $\min_{\mathbf{x}} \mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = D(\boldsymbol{\lambda}, \boldsymbol{\mu})$. In this way, Problem L.2 may be reformulated as the dual optimization problem:

Problem D.1: maximize $D(\boldsymbol{\lambda}, \boldsymbol{\mu})$ over the set of all $\boldsymbol{\lambda}$ and positive $\boldsymbol{\mu}$.

which may be summarized mathematically as follows

$$\begin{aligned} &\max_{\boldsymbol{\lambda},\boldsymbol{\mu}} D(\boldsymbol{\lambda}, \boldsymbol{\mu}) \\ &\text{subject to } \mu_k \geq 0, \ k = 1, ..., p \end{aligned} \tag{D.1}$$

In the context of Problem D.1, $\boldsymbol{\lambda}$ and $\boldsymbol{\mu}$ are also called *dual variables* as they are the arguments of the dual function.

To recap the previous results, there is an equivalence between Problems P.1 and L.1 and between Problems D.1 and L.2. Accordingly, Equation (4) implies that the primal and dual problems may be (approximately) exchanged as per $P.1 \overset{?}{=} D.1$. Section 2.2 will discuss and quantify the error, if any, that is incurred by making this exchange.

The utility of replacing P.1 with D.1 is realized when the number of primal variables m far exceeds the number of dual variables $n + p$. More abstractly, one has the choice of working in either an m or $(n + p)$-dimensional space and then using any number of well-studied optimization techniques (e.g. gradient descent or ascent) to identify $P(\mathbf{x}^*)$ or $D(\boldsymbol{\lambda}^*, \boldsymbol{\mu}^*)$.

2.2 Convexity and Duality

Section 2.1 discussed the means for identifying the stationary points $P(\mathbf{x}^*)$ and $D(\boldsymbol{\lambda}^*, \boldsymbol{\mu}^*)$. However, no mention was made of the non-trivial problem of discerning between a minimum and maximum, nor of the problem of discerning between a local and global optimum. This is why the property of *function convexity* is of great value in constrained optimization. The local optimum of a convex function is also the global optimum, and for this reason much of the literature deals strictly with convex objective functions and constraints for simplicity. However, references such as [9][14][29] demonstrate techniques for transforming non-convex functions to convex ones.

To introduce the fundamentals of convexity, begin with a twice-differentiable function $h(\mathbf{x}) : \mathbb{R}^m \mapsto \mathbb{R}$. The *Hessian* of h, represented by a symmetric matrix $\mathbf{H}(h)$ of size m x m, contains the mixed second derivatives of h as depicted in Equation (7)

$$\mathbf{H}(h) = \begin{bmatrix} \dfrac{\partial^2 h}{\partial x_1^2} & \cdots & \dfrac{\partial^2 h}{\partial x_1 \partial x_i} & \cdots & \dfrac{\partial^2 h}{\partial x_1 \partial x_m} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \dfrac{\partial^2 h}{\partial x_i \partial x_1} & \cdots & \dfrac{\partial^2 h}{\partial x_i^2} & \cdots & \dfrac{\partial^2 h}{\partial x_i \partial x_m} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \dfrac{\partial^2 h}{\partial x_m \partial x_1} & \cdots & \dfrac{\partial^2 h}{\partial x_m \partial x_i} & \cdots & \dfrac{\partial^2 h}{\partial x_m^2} \end{bmatrix} \tag{7}$$

Introducing a column vector $\mathbf{y} \in \mathbb{R}^m$, then the product of $\mathbf{y}^{\mathrm{T}}\mathbf{H}(h)\mathbf{y}$, which is simply a scalar value, may be used to assess the convexity of h:

- If $\mathbf{y}^{\mathrm{T}}\mathbf{H}(h)\mathbf{y} \geq 0$ for every $\mathbf{y} \neq \mathbf{0}^m$ (semi-positive definiteness), then h is convex and a local minimum of h is also a global minimum. However, this global minimum may not be unique.

- If $\mathbf{y}^{\mathrm{T}}\mathbf{H}(h)\mathbf{y} > 0$ for every $\mathbf{y} \neq \mathbf{0}^m$ (positive definiteness), then h is *strictly* convex and a local minimum of h is also a *unique* global minimum.

- If $\mathbf{y}^{\mathrm{T}}\mathbf{H}(h)\mathbf{y} \leq 0$ for every $\mathbf{y} \neq \mathbf{0}^m$ (semi-negative definiteness), then h is concave and a local maximum of h is also a global maximum. However, this global maximum may not be unique.

- If $\mathbf{y}^{\mathrm{T}}\mathbf{H}(h)\mathbf{y} < 0$ for every $\mathbf{y} \neq \mathbf{0}^m$ (negative definiteness), then h is *strictly* concave and a local maximum of h is also a *unique* global maximum.

The four criteria described above may be used to address the conditions for equality between the left and right-hand sides of Equation (4). Although the proof is not repeated here, it has been shown in [8][9][14] that $D(\boldsymbol{\lambda}, \boldsymbol{\mu})$ is concave regardless of

whether $P(\mathbf{x})$ is convex or not; the equivalence of Problems P.1 and D.1 is predicated solely on the behaviour of the primal function.

In the case of $P(\mathbf{x})$ positive or semi-positive definite, equality holds in Equation (4) and there is *strong duality* between the convex primal and concave dual functions. In other words, $P(\mathbf{x}^*) = D(\boldsymbol{\lambda}^*, \boldsymbol{\mu}^*) = \mathcal{L}(\mathbf{x}^*, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*)$ and Problems P.1 and D.1 may be exchanged without any error.

In the more general case of $P(\mathbf{x})$ non-convex, equality does not hold in Equation (4) and there is *weak duality* between the primal and dual functions. The condition of weak duality may be expressed as follows

$$P(\mathbf{x}^*) > \mathcal{L}(\mathbf{x}^*, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*) > D(\boldsymbol{\lambda}^*, \boldsymbol{\mu}^*) \tag{8}$$

In words, the primal minimum and dual maximum bound the saddle point from above and below, respectively. Equation (8) also implies that any non-convex primal problem may be exchanged for the dual problem at the cost of a *duality gap* $d \in \mathbb{R}^+$, or

$$d = P(\mathbf{x}^*) - D(\boldsymbol{\lambda}^*, \boldsymbol{\mu}^*) \tag{9}$$

where d vanishes only under strong duality.

In this thesis, the duality gap concept will be broadened beyond its more traditional definition above. Specifically, d will be generalized to measure the difference between any sub-optimal P and D. Expanding upon this notion, the vectors $\boldsymbol{\epsilon}_x \in \mathbb{R}^m,\ \boldsymbol{\epsilon}_\lambda \in \mathbb{R}^n,$ and $\boldsymbol{\epsilon}_\mu \in \mathbb{R}^p$ are introduced as perturbations applied to the saddle point $(\mathbf{x}^*, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*)$. In this way, $(\widetilde{\mathbf{x}}, \widetilde{\boldsymbol{\lambda}}, \widetilde{\boldsymbol{\mu}}) = (\mathbf{x}^* + \boldsymbol{\epsilon}_x, \boldsymbol{\lambda}^* + \boldsymbol{\epsilon}_\lambda, \boldsymbol{\mu}^* + \boldsymbol{\epsilon}_\mu)$ represents a sub-optimal point in $(m+n+p)$-dimensional space, and the modified duality gap for this point is

$$\widetilde{d} = P(\widetilde{\mathbf{x}}) - D(\widetilde{\boldsymbol{\lambda}}, \widetilde{\boldsymbol{\mu}}) \tag{10}$$

By construction, $\widetilde{d} > d$ for every $(\widetilde{\mathbf{x}}, \widetilde{\boldsymbol{\lambda}}, \widetilde{\boldsymbol{\mu}}) \neq (\mathbf{x}^*, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*)$ and, consequently, inequality (8) may be expanded like so

$$P(\widetilde{\mathbf{x}}) > P(\mathbf{x}^*) > \mathcal{L}(\mathbf{x}^*, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*) > D(\boldsymbol{\lambda}^*, \boldsymbol{\mu}^*) > D(\widetilde{\boldsymbol{\lambda}}, \widetilde{\boldsymbol{\mu}}) \tag{11}$$

In Section 2.5, Equations (10) and (11) will be exploited as a convergence criterion for a duality-inspired numerical algorithm.

2.3 Quadratic Programming

A powerful application of the concepts presented in Sections 2.1 and 2.2 is the solution of continuum mechanics problems. While the objective function and constraints discussed previously were non-linear in general, the equations governing these continua belong to the well-studied QP problem class. The QP class features a primal objective function which is quadratic in its argument with linear equality constraints. Further, the quadratic primal function is *conditionally convex* as will be discussed in greater detail at the end of this chapter.

To best illustrate the link between continuum mechanics and the QP problem class, consider a model thermal diffusion problem on a three-dimensional domain $\Omega \subseteq \mathbb{R}^3$ with closed boundary $\Gamma = \partial\Omega$. The boundary is comprised of two non-overlapping surfaces $\Gamma_{\mathfrak{D}}$ and $\Gamma_{\mathfrak{N}}$ with prescribed Dirichlet temperatures and Neumann fluxes, respectively. The time-interval is taken over the positive real numbers, or $t \in [0, \infty)$. This model problem may be summarized mathematically and graphically as below

$$\begin{aligned}
&-\operatorname{div}(\mathbf{q}(\mathbf{r}, t)) = \rho c_p \frac{\partial u}{\partial t} + S(\mathbf{r}, t), \quad \mathbf{r} \in \Omega,\ t \in [0, \infty) \\
&u(\mathbf{r}, t) = \widetilde{u}(\mathbf{r}, t), \quad \mathbf{r} \in \Gamma_{\mathfrak{D}},\ t \in [0, \infty) \\
&\mathbf{q}(\mathbf{r}, t) = \widetilde{\mathbf{q}}(\mathbf{r}, t), \quad \mathbf{r} \in \Gamma_{\mathfrak{N}},\ t \in [0, \infty) \\
&u(\mathbf{r}, t) = u_0(\mathbf{r}), \quad \mathbf{r} \in \Omega,\ t = 0
\end{aligned} \tag{12}$$

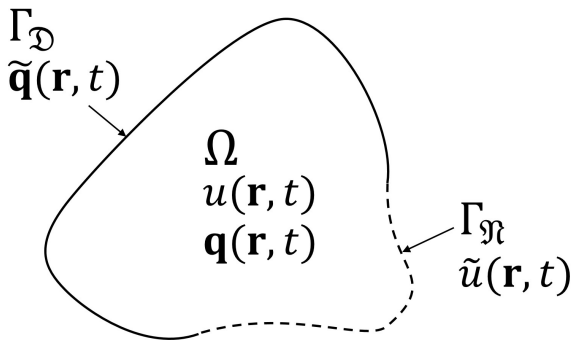


Figure 1: A two-dimensional domain with the Dirichlet sub-boundary ($\Gamma_{\mathfrak{D}}$, solid line) and Neumann sub-boundary ($\Gamma_{\mathfrak{N}}$, dashed line) shown. The Dirichlet temperature and Neumann flux BCs are also displayed.

where the vector flux field $\mathbf{q}(\mathbf{r},t)$ and scalar temperature field $u(\mathbf{r},t)$ are the variables of interest and are functions of *spacetime*[4]. Functions of spacetime are indicated by the arguments $(\mathbf{r},t)$, which will henceforth be suppressed when there is no risk of ambiguity.

In Equation (12), S is the volumetric source/sink density field, $\widetilde{u}$ is the Dirichlet temperature, and $\widetilde{\mathbf{q}}$ is the Neumann flux, all of which are prescribed functions of spacetime. The initial temperature field u_0 is a prescribed function of space and the density, thermal conductivity and specific heat capacity $\rho,\ \kappa,$ and c_p are isotropic, positive scalars.

Consider two function spaces $\mathscr{Q}$ and $\mathscr{U}$ which are defined as follows:

- $\mathscr{Q}$ contains all smooth flux fields defined on the spacetime domain $(\Omega \cup \Gamma)$ x $[0,\infty)$. $\mathscr{Q}$ is spanned by an infinite number of basis functions $\psi^q(\mathbf{r},t)$ and any flux field $\mathbf{q} \in \mathscr{Q}$ may be expressed as a linear combination of these ψ^q.

- $\mathscr{U}$ contains all smooth temperature fields defined on the spacetime domain. $\mathscr{U}$ is spanned by an infinite number of basis functions $\psi^u(\mathbf{r},t)$ and any temperature field $u \in \mathscr{U}$ may be expressed as a linear combination of these ψ^u.

[4]Note that the concept of spacetime in this thesis differs slightly from that encountered in relativistic mechanics (i.e. Minkowski space).

Further, $\mathscr{Q}$ and $\mathscr{U}$ are restricted to contain only the flux and temperature fields which satisfy the Neumann and Dirichlet boundary conditions, respectively, as well as the initial conditions.

The primal QP problem may now be formally introduced; a flux field $\mathbf{q}^* \in \mathscr{Q}$ is sought which minimizes a potential energy functional[5], $P[\mathbf{q}] : \mathscr{Q} \mapsto \mathbb{R}$, subject to the heat equation in (12)

$$\begin{aligned} &\underset{\mathbf{q}\in\mathscr{Q}}{\text{minimize}} \quad P[\mathbf{q}] = \int_0^\infty \int^\Omega \frac{\kappa^{-1}}{2}\mathbf{q}^2 dV dt \\ &\text{subject to} \;\; \operatorname{div}(\mathbf{q}) + \rho c_p \frac{\partial u}{\partial t} + S = 0 \end{aligned} \tag{13}$$

Following suit with Section 2.1, the Lagrangian functional unifies $P[\mathbf{q}]$ with the constraints

$$\mathcal{L}[\mathbf{q}, u] = \int_0^\infty \left[\int^\Omega \frac{\kappa^{-1}}{2}\mathbf{q}^2 dV - \int^\Omega u \left(\operatorname{div}(\mathbf{q}) + \rho c_p \frac{\partial u}{\partial t} + S \right) dV \right] dt \tag{14}$$

To summarize the ideas introduced in Equations (13) and (14), reference is made back to Problem (P.1) and Equation (1) with the following observations:

- $P[\mathbf{q}]$ is the functional analog of the primal function, $P(\mathbf{x})$.
- The flux fields $\mathbf{q} \in \mathscr{Q}$ are the continuous analogs of the primal variables, $\mathbf{x}$.
- The temperature fields $u \in \mathscr{U}$ are the continuous analogs of the Lagrange multipliers/dual variables, $\boldsymbol{\lambda}$.
- u penalizes the subset of flux fields in $\mathscr{Q}$ which violate the heat equation. In this way, the second integral on the right-hand side of Equation (14) represents the energy contained in the residual field, $R = \operatorname{div}(\mathbf{q}) + \rho c_p \partial u/\partial t + S$.

[5] A functional maps a function, or spacetime field in this context, to a scalar real number.

To simplify the forthcoming analysis of Equation (14), the term containing $u\text{div}(\mathbf{q})$ is transformed using integration by parts

$$\int^{\Omega} u\text{div}(\mathbf{q})dV = -\int^{\Omega} \text{grad}(u)\cdot\mathbf{q}dV + \int^{\Gamma_{\mathfrak{D}}} \widetilde{u}\mathbf{q}\cdot d\mathbf{a} + \int^{\Gamma_{\mathfrak{N}}} u\widetilde{\mathbf{q}}\cdot d\mathbf{a} \qquad (15)$$

where $d\mathbf{a}$ represents a surface of infinitesimal area da with an outward-oriented normal $\mathbf{n}$, such that $d\mathbf{a} = \mathbf{n}da$. Substitution of Equation (15) into (14) gives

$$\mathcal{L}[\mathbf{q},u] = \int_0^{\infty}\left[\int^{\Omega}\frac{\kappa^{-1}}{2}\mathbf{q}^2dV + \int^{\Omega}\text{grad}(u)\cdot\mathbf{q}dV - \int^{\Omega} u\left(\rho c_p\frac{\partial u}{\partial t} + S\right)dV - \int^{\Gamma_{\mathfrak{D}}}\widetilde{u}\mathbf{q}\cdot d\mathbf{a} - \int^{\Gamma_{\mathfrak{N}}} u\widetilde{\mathbf{q}}\cdot d\mathbf{a}\right]dt \qquad (16)$$

This alternate form of the Lagrangian functional, referred to in the mixed FEM literature as the *Hellinger-Reissner form* [16][30][31][32], will be used henceforth.

It is possible to find the optimal field pair $(\mathbf{q}^*, u^*)$ using the calculus of variations; this is the approach taken in Lagrangian and Hamiltonian mechanics, which themselves bear strong resemblance to the QP methodology presented here [33][34]. Rather, in fitting with the finite-dimensional analyses in Sections 4.1 and 4.2, the discrete form of Equation (16) is sought instead. This alternative has the benefit of converting the constraint from a partial differential equation to a SLE.

The first step in the discretization involves interpolation of $\mathbf{q}$ and u onto two spacetime grids of finite size. The temporal domain is discretized with τ time-steps of uniform size Δt, such that $t_k = k\Delta t$ with $k = 0, ..., \tau$. The interpolation scheme in the spatial domain will be left general for the moment, with two different techniques being explored in Sections 4.1 and 4.2. It is sufficient for now to discretize the spatial domain with the following staggered grids:

Primal Grid - M nodes total

- m nodes on $\Omega \cup \Gamma_{\mathfrak{D}}$ labelled $1, ..., m$; contain the free fluxes.
- $\widetilde{m} = M - m$ Neumann nodes on $\Gamma_{\mathfrak{N}}$ labelled $m+1, ..., M$; contain the prescribed Neumann fluxes.

Dual Grid - N nodes total

- $\widetilde{n} = N - n$ Dirichlet nodes on $\Gamma_{\mathfrak{D}}$ labelled $1, ..., \widetilde{n}$; contain the prescribed Dirichlet temperatures.
- n nodes on $\Omega \cup \Gamma_{\mathfrak{N}}$ labelled $\widetilde{n} + 1, ..., N$; contain the free temperatures.

These spatial grids are so named for the respective flux (primal) and temperature (dual) fields defined on them. To clarify the abstract details outlined above, refer to Figure 2 for a simple example. Note that the "free" fluxes/temperatures mentioned above are those that are to be computed with SLE (22) and are not prescribed as BCs.

Mathematically, discretization of space and time constitutes a projection of the function spaces $\mathscr{Q}$ and $\mathscr{U}$ onto the finite-dimensional vector spaces $\mathscr{Q}_h$ and $\mathscr{U}_h$. These vector spaces are respectively spanned by a set of $m(\tau+1)$ and $n(\tau+1)$ basis functions in spacetime, each of the form $\psi^{q,u}(\mathbf{r} - \mathbf{r}_i, t - t_k)$ with $i = 1, ..., M/N$ and $k = 0, ..., \tau$. The $(\mathbf{r} - \mathbf{r}_i, t - t_k)$ convention indicates that the $(i,k)^{th}$ basis function uses a local coordinate system with its origin at $\mathbf{r}_i$ in space and t_k in time.

For the sake of simplification, it is assumed *a priori* that ψ^q and ψ^u are both separable into space and time components. Consequently, $\psi^{q,u}(\mathbf{r} - \mathbf{r}_i, t - t_k)$ may be expressed as a product of *space-like* and *time-like* basis functions, or $\phi^{q,u}(\mathbf{r} - \mathbf{r}_i)\theta(t - t_k),\ \forall i, k$. By making the abbreviations $\phi^{q,u}(\mathbf{r} - \mathbf{r}_i) = \phi_i^{q,u}(\mathbf{r})$ and $\theta(t - t_k) = \theta_k(t)$, the flux and temperature may be approximated at any $\mathbf{r} \in \Omega$ and $t \in [t_0, t_\tau]$ by the following vector-matrix products

$$
\begin{aligned}
\mathbf{q}(\mathbf{r},t) \approx \boldsymbol{\phi}^q \mathbf{Q} \boldsymbol{\theta} &= \begin{bmatrix} \phi_1^q(\mathbf{r}) \\ \vdots \\ \phi_i^q(\mathbf{r}) \\ \vdots \\ \phi_m^q(\mathbf{r}) \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} \mathbf{q}_{1,0} & \cdots & \mathbf{q}_{1,k} & \cdots & \mathbf{q}_{1,\tau} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \mathbf{q}_{i,0} & \cdots & \mathbf{q}_{i,k} & \cdots & \mathbf{q}_{i,\tau} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \mathbf{q}_{m,0} & \cdots & \mathbf{q}_{m,k} & \cdots & \mathbf{q}_{m,\tau} \end{bmatrix} \begin{bmatrix} \theta_0(t) \\ \vdots \\ \theta_k(t) \\ \vdots \\ \theta_\tau(t) \end{bmatrix} \\
u(\mathbf{r},t) \approx \boldsymbol{\phi}^u \mathbf{U} \boldsymbol{\theta} &= \begin{bmatrix} \phi_{\widetilde{n}+1}^u(\mathbf{r}) \\ \vdots \\ \phi_{\widetilde{n}+i}^u(\mathbf{r}) \\ \vdots \\ \phi_N^u(\mathbf{r}) \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} u_{\widetilde{n}+1,0} & \cdots & u_{\widetilde{n}+1,k} & \cdots & u_{\widetilde{n}+1,\tau} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ u_{\widetilde{n}+i,0} & \cdots & u_{\widetilde{n}+i,k} & \cdots & u_{\widetilde{n}+i,\tau} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ u_{N,0} & \cdots & u_{N,k} & \cdots & u_{N,\tau} \end{bmatrix} \begin{bmatrix} \theta_0(t) \\ \vdots \\ \theta_k(t) \\ \vdots \\ \theta_\tau(t) \end{bmatrix}
\end{aligned} \tag{17}
$$

where $\boldsymbol{\phi}^q$ and $\boldsymbol{\phi}^u$ are row vectors of length m and n containing the set of space-like basis functions. $\boldsymbol{\theta}$ is a column vector of length $\tau + 1$ containing the set of time-like basis functions. The matrices $\mathbf{Q} \in \mathscr{Q}_h$ and $\mathbf{U} \in \mathscr{U}_h$ represent the free, discrete spacetime flux and temperature fields, respectively, and the shorthand $\square_{i,k}$ indicates the i^{th} node and k^{th} time-step. Equation (17) may alternatively be represented by a pair of finite linear combinations over spacetime, or $\mathbf{q}(\mathbf{r},t) \approx \sum_{i=1}^{m} \sum_{k=0}^{\tau} \mathbf{q}_{i,k} \phi_i^q(\mathbf{r}) \theta_k(t)$ and $u(\mathbf{r},t) \approx \sum_{i=\widetilde{n}+1}^{N} \sum_{k=0}^{\tau} u_{i,k} \phi_i^u(\mathbf{r}) \theta_k(t)$.

However, the formulae above are valid only for $\mathbf{r} \in \Omega$. Rather, the flux and temperature may be approximated at any $\mathbf{r} \in \Gamma$ by the following vector-matrix products

$$\widetilde{\mathbf{q}}(\mathbf{r},t) = \underbrace{\begin{bmatrix} \phi^q_{m+1}(\mathbf{r}) \\ \vdots \\ \phi^q_{m+i}(\mathbf{r}) \\ \vdots \\ \phi^q_{M}(\mathbf{r}) \end{bmatrix}^{\mathrm{T}}}_{=\widetilde{\boldsymbol{\phi}}^q} \underbrace{\begin{bmatrix} \widetilde{\mathbf{q}}(\mathbf{r}_{m+1},t_0) & \cdots & \widetilde{\mathbf{q}}(\mathbf{r}_{m+1},t_k) & \cdots & \widetilde{\mathbf{q}}(\mathbf{r}_{m+1},t_\tau) \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \widetilde{\mathbf{q}}(\mathbf{r}_{m+i},t_0) & \cdots & \widetilde{\mathbf{q}}(\mathbf{r}_{m+i},t_k) & \cdots & \widetilde{\mathbf{q}}(\mathbf{r}_{m+i},t_\tau) \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \widetilde{\mathbf{q}}(\mathbf{r}_{M},t_0) & \cdots & \widetilde{\mathbf{q}}(\mathbf{r}_{M},t_k) & \cdots & \widetilde{\mathbf{q}}(\mathbf{r}_{M},t_\tau) \end{bmatrix}}_{=\widetilde{\mathbf{Q}}} \begin{bmatrix} \theta_0(t) \\ \vdots \\ \theta_k(t) \\ \vdots \\ \theta_\tau(t) \end{bmatrix}$$

$$\widetilde{u}(\mathbf{r},t) = \underbrace{\begin{bmatrix} \phi^u_{1}(\mathbf{r}) \\ \vdots \\ \phi^u_{i}(\mathbf{r}) \\ \vdots \\ \phi^u_{\widetilde{n}}(\mathbf{r}) \end{bmatrix}^{\mathrm{T}}}_{=\widetilde{\boldsymbol{\phi}}^u} \underbrace{\begin{bmatrix} \widetilde{u}(\mathbf{r}_{1},t_0) & \cdots & \widetilde{u}(\mathbf{r}_{1},t_k) & \cdots & \widetilde{u}(\mathbf{r}_{1},t_\tau) \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \widetilde{u}(\mathbf{r}_{i},t_0) & \cdots & \widetilde{u}(\mathbf{r}_{i},t_k) & \cdots & \widetilde{u}(\mathbf{r}_{i},t_\tau) \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \widetilde{u}(\mathbf{r}_{\widetilde{n}},t_0) & \cdots & \widetilde{u}(\mathbf{r}_{\widetilde{n}},t_k) & \cdots & \widetilde{u}(\mathbf{r}_{\widetilde{n}},t_\tau) \end{bmatrix}}_{=\widetilde{\mathbf{U}}} \begin{bmatrix} \theta_0(t) \\ \vdots \\ \theta_k(t) \\ \vdots \\ \theta_\tau(t) \end{bmatrix} \tag{18}$$

where $\widetilde{\boldsymbol{\phi}}^q$ and $\widetilde{\boldsymbol{\phi}}^u$ are row vectors of length $\widetilde{m}$ and $\widetilde{n}$, respectively. Specifically, $\widetilde{\boldsymbol{\phi}}^q$ and $\widetilde{\boldsymbol{\phi}}^u$ contain the set of space-like basis functions centered on Neumann nodes $\mathbf{r}_{m+1},...,\mathbf{r}_M \in \Gamma_{\mathfrak{N}}$ and Dirichlet nodes $\mathbf{r}_1,...,\mathbf{r}_{\widetilde{n}} \in \Gamma_{\mathfrak{D}}$. Recall that the boundary functions $\widetilde{\mathbf{q}}(\mathbf{r},t)$ and $\widetilde{u}(\mathbf{r},t)$ are prescribed as per Equation (12).

Appendix A contains a derivation of the Hellinger-Reissner Lagrangian functional which is discrete in space and time. To summarize, the vector-matrix products in (17) and (18) are substituted into Equation (16) and, following a sequence of mathematical manipulations, the KKT conditions are imposed upon the resulting discrete Lagrangian function. The output of this step is a SLE of size m x τ for the flux field

$\mathbf{Q}$ and a SLE of size n x τ for the temperature field $\mathbf{U}$. These two systems are represented by Equations (112) and (114) and, before they are combined, the following substitutions are made to better reflect the QP literature

$$\begin{aligned} \mathbf{C}^{-1} &= \kappa^{-1}\mathbf{G}(\boldsymbol{\phi}^q, \boldsymbol{\phi}^q, \Omega) \\ \mathbf{A} &= \mathbf{G}(\boldsymbol{\phi}^q, \mathrm{grad}(\boldsymbol{\phi}^u), \Omega) \\ \mathbf{M} &= \frac{\rho c_p}{\Delta t}\mathbf{G}(\boldsymbol{\phi}^u, \boldsymbol{\phi}^u, \Omega) \end{aligned} \tag{19}$$

where the matrices labelled $\mathbf{G}(\boldsymbol{\phi}^{\square}, \boldsymbol{\phi}^{\square}, \square)$ indicate *Gramian matrices* which, informally, describe the "overlap" in space between each pair of basis functions $\phi_i^{\square}(\mathbf{r})$ and $\phi_j^{\square}(\mathbf{r})$. See Equations (97) and (98) for details concerning the structure and labelling conventions for these Gramian matrices.

The problem data specified in Equation (12) are amalgamated into the two matrices defined below

$$\begin{aligned} \mathbf{F}_1 &= \mathbf{G}(\boldsymbol{\phi}^q, \widetilde{\boldsymbol{\phi}}^u, \Gamma_{\mathfrak{D}})\widetilde{\mathbf{U}} \\ \mathbf{F}_2 &= \frac{\rho c_p}{\Delta t}\mathbf{G}(\boldsymbol{\phi}^u, \boldsymbol{\phi}^u, \Omega)\mathbf{U}^{\dagger} - \mathbf{G}(\boldsymbol{\phi}^u, \boldsymbol{\phi}^u, \Omega)\mathbf{S} - \mathbf{G}(\boldsymbol{\phi}^u, \widetilde{\boldsymbol{\phi}}^q, \Gamma_{\mathfrak{N}})\widetilde{\mathbf{Q}} \end{aligned} \tag{20}$$

where $\mathbf{S}$ is an n x τ matrix representing the volumetric source/sink field in discrete spacetime; it has entries $\mathbf{S}_{i,k} = S(\mathbf{r}_i, t_k)$. $\mathbf{U}^{\dagger}$ is an n x τ matrix representing the "temperature lag" at each time-step (see Equation (25) for details).

In this standard QP notation, the discrete Hellinger-Reissner Lagrangian function signified by (110) becomes

$$\mathcal{L}(\mathbf{Q}, \mathbf{U}) = \mathrm{tr}\left[\frac{1}{2}\mathbf{Q}^{\mathrm{T}}\mathbf{C}^{-1}\mathbf{Q} - \mathbf{F}_1^{\mathrm{T}}\mathbf{Q} - \mathbf{U}^{\mathrm{T}}(-\mathbf{A}^{\mathrm{T}}\mathbf{Q} + \mathbf{M}\mathbf{U} - \mathbf{F}_2)\right]\Delta t \tag{21}$$

Likewise, SLEs (112) and (114) may be consolidated into a SLE of size $(m+n)$ x τ, the

solution of which amounts to finding the saddle point of (21). In this way, the superscript $\square^*$ is reintroduced to indicate the flux-temperature fields which simultaneously satisfy (112) and (114)

$$\begin{bmatrix} \mathbf{C}^{-1} & \mathbf{A} \\ -\mathbf{A}^{\mathrm{T}} & \mathbf{M} \end{bmatrix} \begin{bmatrix} \mathbf{Q}^* \\ \mathbf{U}^* \end{bmatrix} = \begin{bmatrix} \mathbf{F}_1 \\ \mathbf{F}_2 \end{bmatrix} \tag{22}$$

This SLE lies at the heart of the relatively novel field of mixed finite element methods [16]. The "mixed" terminology emphasizes the equal importance of the primal and dual solutions. Conversely, standard FEM typically solves only for the dual temperatures, with the primal fluxes following by post-processing (although the primal/dual solution concept is almost never explicitly addressed in the FEM literature).

More importantly, SLE (22) inherits a unique meaning depending on the continuum being considered. In particular, $\mathbf{C}^{-1}\mathbf{Q}^* + \mathbf{A}\mathbf{U}^* = \mathbf{F}_1$ is the discrete analog of Fourier's law $\kappa^{-1}\mathbf{q} + \mathrm{grad}(u) = 0$ and generalizes to a *constitutive law* for other continua (e.g. Hooke's law in solid mechanics and Ohm's law in electric circuits). In like manner, $-\mathbf{A}^{\mathrm{T}}\mathbf{Q}^* + \mathbf{M}\mathbf{U}^* = \mathbf{F}_2$ is the discrete analog of the heat equation $-\mathrm{div}(\mathbf{q}) = \rho c_p \partial u/\partial t + S$ and generalizes to a *conservation law* for other continua (e.g. the quasi-static conservation of momentum equation in solid mechanics and Kirchoff's law in electric circuits). Further, each of the symbols in SLE (22) take on an intuitive, universal meaning

- $\mathbf{Q}^*$ generalizes to the spacetime field of primal variables.
- $\mathbf{U}^*$ generalizes to the spacetime field of Lagrange multipliers/dual variables.
- $\mathbf{C}^{-1}$ is the square, symmetric matrix of material constants.
- $\mathbf{A}$ is the discrete gradient operator and $\mathbf{A}^{\mathrm{T}}$ is the discrete divergence operator.

- $\mathbf{M}$ is the mass matrix whose form depends on the time-discretization scheme ($\mathbf{M} = \mathbf{0}^{nxn}$ for continua at steady-state or in quasi-static equilibrium).
- $\mathbf{F}_1$ is the matrix representing the effects of the Dirichlet boundary conditions.
- $\mathbf{F}_2$ is the matrix representing the unsteady term, volumetric source/sink magnitudes, and the effects of the Neumann boundary conditions.

To conclude this section, the BCs in the QP problem will be briefly discussed. Since the primal $\mathbf{Q}^*$ and dual $\mathbf{U}^*$ both appear in Equation (22), the Neumann fluxes and Dirichlet temperatures are both, in a sense, essential BCs. That is, the Neumann BCs are enforced through the $\mathbf{F}_2$ matrix in $-\mathbf{A}^{\mathrm{T}}\mathbf{Q}^* + \mathbf{M}\mathbf{U}^* = \mathbf{F}_2$ and the Dirichlet BCs are enforced through the $\mathbf{F}_1$ matrix in $\mathbf{C}^{-1}\mathbf{Q}^* + \mathbf{A}\mathbf{U}^* = \mathbf{F}_1$. This interpretation of the BCs will be made more specific in Sections 4.1 and 4.2.

2.4 The LATIN Algorithm

In general, SLE (22) may be solved using segregated or coupled means [15]. The segregated solution techniques first transform from the $(m+n)$ x τ-dimensional space embedding the Lagrangian to either the primal or dual variable space, then optimize only the variables in the selected space. Thus, these solution methods have the advantage of operating in a lower dimension (i.e. m x τ in the primal space and n x τ in the dual), the implication being that significant computational expense may be saved for large problems. Following maximization of the dual variables (for example), the corresponding minimal primal variables may then be computed by elimination from SLE (22). The coupled solution techniques solve (22) for the primal and dual variables simultaneously; they can become prohibitively expensive for large problems.

Further, both the segregated and coupled methods may be solved directly or iteratively. One such iterative, segregated solution technique is the LATIN method [11][12]. More specifically, LATIN is a predictor-corrector technique which borrows from duality and Reduced Order Modelling (ROM) to converge to an exact discrete solution. To make the following discussion more tangible, the algorithm will be introduced in the context of a one-dimensional, transient thermal diffusion test problem. The geometry of this test problem is presented in Figure 2

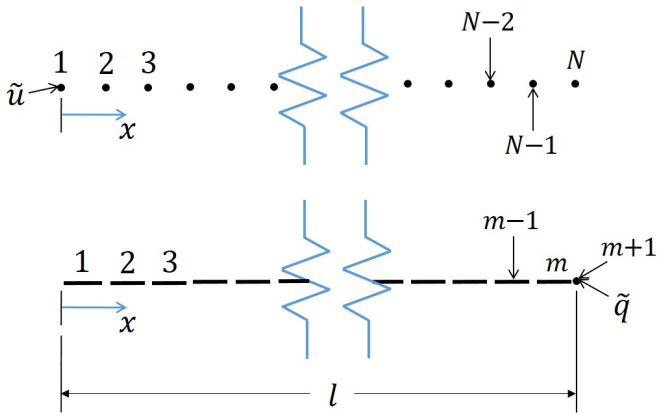


Figure 2: One-dimensional domain of length l with the grid numbering scheme specified on page 35. The dual grid (top) consists of N nodes, with $\widetilde{n} = 1$ Dirichlet node labelled 1 and $n = N - \widetilde{n}$ free nodes labelled 2 through N. Likewise, the primal grid (bottom) consists of M line segments connecting adjacent nodes, with m free lines labelled 1 through m and $\widetilde{m} = 1$ Neumann surface labelled $m + 1$.

Spatial discretization consists of two staggered grids as in Figure 2, with the m line elements comprising the primal grid and the n nodes comprising the dual grid. Temporal discretization is done with a backward Euler scheme and a time interval $t = t_0, ..., t_\tau$.

The problem consists of the PDE and boundary/initial conditions as follows

$$
\begin{aligned}
-\operatorname{div}(q) &= \rho c_p \frac{\partial u}{\partial t},\ x \in [0,l],\ t \in [t_0, t_\tau] \\
u(x=0,t) &= \widetilde{u}(t),\ t \in [t_0, t_\tau] \\
q(x=l,t) &= \widetilde{q}(t),\ t \in [t_0, t_\tau] \\
u(x,t=0) &= u_0(x),\ x \in [0,l]
\end{aligned}
\tag{23}
$$

The left end of the domain is subjected to a Dirichlet boundary temperature $\widetilde{u}(t)$ and the right end (technically a surface of unit area) is subjected to a Neumann boundary flux $\widetilde{q}(t)$. An initial temperature field $u_0(x)$ is prescribed along with a uniform density ρ, thermal conductivity κ, and specific heat capacity c_p. Therefore, these thermal properties, which are generally dependent on temperature, are simplified from relatively complicated tensors to scalar values. The matrices $\mathbf{Q}$ and $\mathbf{U}$ in Equation (17) provide a natural means for representing the primal flux and dual temperature fields in spacetime. Note that the first column of each matrix is deleted as explained on page 154.

Recall from Section 2.3 the function spaces $\mathscr{Q}$ and $\mathscr{U}$ and the associated continuous spacetime fields ($\mathbf{q}^* \in \mathscr{Q}, u^* \in \mathscr{U}$). As a consequence of the convexity of the QP problem class, this unique pair possesses the following properties:

- it lies at the saddle point of the **continuous** Lagrangian functional (16).
- it satisfies **strong** duality as defined on page 30.

Analogous to $\mathscr{Q}$ and $\mathscr{U}$ are the finite-dimensional vector spaces $\mathscr{Q}_h$ and $\mathscr{U}_h$ which were also defined in Section 2.3. The pair of discrete spacetime fields ($\mathbf{Q}^* \in \mathscr{Q}_h, \mathbf{U}^* \in \mathscr{U}_h$) possesses the following properties:

- it lies at the saddle point of the **discrete** Lagrangian function (21).
- it satisfies **weak** duality as per Equation (8).

Equivalently, if Equation (22) is relaxed to include all sub-optimal pairs ($\mathbf{Q} \in \mathscr{Q}_h, \mathbf{U} \in \mathscr{U}_h$),

$$\begin{bmatrix} \mathbf{C}^{-1} & \mathbf{A} \\ -\mathbf{A}^{\mathrm{T}} & \mathbf{M} \end{bmatrix} \begin{bmatrix} \mathbf{Q} \\ \mathbf{U} \end{bmatrix} - \begin{bmatrix} \mathbf{F}_1 \\ \mathbf{F}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{R}_1 \\ \mathbf{R}_2 \end{bmatrix} \tag{24}$$

then $(\mathbf{Q}^*, \mathbf{U}^*)$ simultaneously minimizes the residuals in Fourier's law and the heat equation, or $\mathbf{R}_1$ and $\mathbf{R}_2$ respectively. To summarize, the pair of infinite-dimensional fields $(\mathbf{q}^*, u^*)$ represents the *exact solution* whereas the pair of finite-dimensional fields $(\mathbf{Q}^*, \mathbf{U}^*)$ represents an *approximation to the exact solution*; LATIN seeks to converge iteratively to the latter. Recall that $\mathbf{F}_1$ and $\mathbf{F}_2$, which are governed by Equation (20), encapsulate the given problem data and are therefore matrices of known values.

As a first step, LATIN segregates the $(m + n)$ x τ-dimensional space embedding the Lagrangian into four manifolds. Subsections 2.4.1 and 2.4.2 will discuss these manifolds in detail as well as how each one is linked.

2.4.1 Kinematically Admissible Manifolds

A pair of KA manifolds is defined below. The KA terminology is borrowed from solid mechanics where it refers to the set of displacement and strain fields that satisfy the Dirichlet and compatibility conditions. The thermal diffusion test problem considered here is nearly analogous, and the flux and temperature fields contained in the two KA manifolds will henceforth be indicated by a hat $\widehat{\square}$:

- $\mathscr{K}_D \subset \mathbb{R}^{n\mathrm{x}\tau}$ - Contains all n x τ dual temperature fields which satisfy the Dirichlet BC $\widetilde{u}(t)$.
- $\mathscr{K}_P \subset \mathbb{R}^{m\mathrm{x}\tau}$ - Contains all m x τ primal flux fields which satisfy Fourier's law, or in set-builder notation $\{\exists \widehat{\mathbf{Q}} \in \mathscr{K}_P, \forall \widehat{\mathbf{U}} \in \mathscr{K}_D \mid \mathbf{C}^{-1}\widehat{\mathbf{Q}} + \mathbf{A}\widehat{\mathbf{U}} = \mathbf{F}_1\}$. Alternatively, $\mathscr{K}_P$ is the image of $\mathscr{K}_D$ under Fourier's law.

Therefore, a KA flux-temperature field pair simultaneously satisfies the Dirichlet BC and Fourier's law, such that $(\widehat{\mathbf{Q}}, \widehat{\mathbf{U}}) \in (\mathscr{K}_P \cup \mathscr{K}_D)$.

The LATIN algorithm is initialized with a user-defined trial solution, labelled $\widehat{\mathbf{U}}^0 \in \mathscr{K}_D$, for the KA spacetime temperature field. Moreover, the selection of $\widehat{\mathbf{U}}^0$ determines the entries of its corresponding temperature lag matrix $(\widehat{\mathbf{U}}^\dagger)^0$ as in Equation (109). Specifically,

$$(\widehat{\mathbf{U}}^\dagger)^0 = \begin{bmatrix} u_{2,0} & \widehat{u}^0_{2,1} & \cdots & \widehat{u}^0_{2,k-1} & \cdots & \widehat{u}^0_{2,\tau-1} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ u_{i,0} & \widehat{u}^0_{i,1} & \cdots & \widehat{u}^0_{i,k-1} & \cdots & \widehat{u}^0_{i,\tau-1} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ u_{N,0} & \widehat{u}^0_{N,1} & \cdots & \widehat{u}^0_{N,k-1} & \cdots & \widehat{u}^0_{N,\tau-1} \end{bmatrix} \tag{25}$$

where the first column of $(\widehat{\mathbf{U}}^\dagger)^0$ is constrained to match the initial condition. The matrices $\widetilde{\mathbf{U}}$ and $\widetilde{\mathbf{Q}}$, themselves contained within respective matrices $\mathbf{F}_1$ and $\mathbf{F}_2$, enforce the Dirichlet and Neumann BCs per Equation (18).

From $\widehat{\mathbf{U}}^0$, the KA flux field $\widehat{\mathbf{Q}}^0 \in \mathscr{K}_P$ is calculated via Fourier's law, $\mathbf{C}^{-1}\widehat{\mathbf{Q}}^0 + \mathbf{A}\widehat{\mathbf{U}}^0 = \mathbf{F}_1$. This step has the effect of projecting from the dual KA manifold to the primal KA manifold.

The residual fields are now sought by substitution of the KA primal-dual pair $(\widehat{\mathbf{Q}}^0, \widehat{\mathbf{U}}^0)$ into Equation (24). By definition, $\mathbf{R}_1^0$ vanishes because Fourier's law holds exactly for all KA flux-temperature field pairs. The residuals in the heat equation follow by elimination of $\widehat{\mathbf{Q}}^0$ from Equation (24); row one is left-multiplied by $\mathbf{A}^{\mathrm{T}}\mathbf{C}$ and the result is added to row two[6]

[6]The $\mathbf{C}^{-1}$ matrix is assumed for now to be well-conditioned and invertible.

$$(\mathbf{M} + \mathbf{A}^{\mathrm{T}}\mathbf{C}\mathbf{A})\widehat{\mathbf{U}}^0 - (\mathbf{F}_2 + \mathbf{A}^{\mathrm{T}}\mathbf{C}\mathbf{F}_1) = \mathbf{R}_2^0 \tag{26}$$

with $\mathbf{R}_2^0$ being stored for later use.

2.4.2 Statically Admissible Manifolds

A pair of SA manifolds is defined below. The SA terminology is borrowed from solid mechanics where it refers to the set of stress fields that satisfy quasi-static equilibrium and the Neumann BCs. Since the problem at hand is transient and thus has a non-zero $\mathbf{M}$ matrix, the definition of SA must be appropriately modified. The flux and temperature fields contained in the two SA manifolds will henceforth be indicated by a bar $\overline{\square}$:

- $\mathscr{S}_P \subset \mathbb{R}^{m\mathrm{x}\tau}$ - Contains all m x τ primal flux fields which satisfy the Neumann BC $\widetilde{q}(t)$.
- $\mathscr{S}_D \subset \mathbb{R}^{n\mathrm{x}\tau}$ - Contains all n x τ dual temperature fields which satisfy the heat equation, or in set-builder notation $\{\exists\overline{\mathbf{U}} \in \mathscr{S}_D, \forall\overline{\mathbf{Q}} \in \mathscr{S}_P \mid -\mathbf{A}^{\mathrm{T}}\overline{\mathbf{Q}} + \mathbf{M}\overline{\mathbf{U}} = \mathbf{F}_2\}$. Alternatively, $\mathscr{S}_D$ is the image of $\mathscr{S}_P$ under the heat equation.

Therefore, a SA flux-temperature field pair simultaneously satisfies the Neumann BC and heat equation, such that $(\overline{\mathbf{Q}}, \overline{\mathbf{U}}) \in (\mathscr{S}_P \cup \mathscr{S}_D)$.

Beginning with $\widehat{\mathbf{Q}}^0$, the SA flux $\overline{\mathbf{Q}}^{1/2} \in \mathscr{S}_P$ is approximated by two successive spatial interpolations. $\widehat{\mathbf{Q}}^0$, which is defined on the m line segments, is first interpolated on to the N nodes and the output is labelled $\widehat{\mathbf{Q}}^{1/2}$. It is important to note that, in general, $\widehat{\mathbf{Q}}^{1/2}$ will not satisfy the Neumann flux at $x_N = l$. $\overline{\mathbf{Q}}^{1/2}$ follows from interpolation of $\widehat{\mathbf{Q}}^{1/2}$ back on to the m line centroids **with incorporation of the Neumann flux**. Both interpolation steps are accomplished via Equation (17)

and depend on the selection of $\boldsymbol{\phi}^q$; Sections 4.1 and 4.2 will explore this facet of the LATIN algorithm. These details notwithstanding, the two interpolations have the collective effect of projecting from $\mathscr{K}_P$ on to $\mathscr{S}_P$.

We now seek the SA temperature field and assume *a priori* that this field is KA as well. In a geometric sense, this unique temperature field $\widehat{\mathbf{U}}^1$ lies at the intersection of the $\mathscr{K}_D$ and $\mathscr{S}_D$ manifolds and thus represents an updated guess for $\mathbf{U}^*$. Consequently, substitution of $(\overline{\mathbf{Q}}^{1/2}, \widehat{\mathbf{U}}^1)$ into Equation (24) causes both residual fields $\mathbf{R}_1^1$ and $\mathbf{R}_2^1$ to vanish. This substitution may be simplified by elimination of $\overline{\mathbf{Q}}^{1/2}$, as was done in Equation (26)

$$(\mathbf{M} + \mathbf{A}^{\mathrm{T}}\mathbf{C}\mathbf{A})\widehat{\mathbf{U}}^1 - (\mathbf{F}_2 + \mathbf{A}^{\mathrm{T}}\mathbf{C}\mathbf{F}_1) = \mathbf{0}^{n\mathrm{x}\tau} \tag{27}$$

An additional substitution $\widehat{\mathbf{U}}^1 = \widehat{\mathbf{U}}^0 + \mathbf{D}^1$ is now made, where $\mathbf{D}^1$ is the n x τ temperature correction field for the first LATIN iteration. Implementing this substitution into (27) and rearranging the resulting expression yields the following

$$[(\mathbf{M} + \mathbf{A}^{\mathrm{T}}\mathbf{C}\mathbf{A})\widehat{\mathbf{U}}^0 - (\mathbf{F}_2 + \mathbf{A}^{\mathrm{T}}\mathbf{C}\mathbf{F}_1)] + (\mathbf{M} + \mathbf{A}^{\mathrm{T}}\mathbf{C}\mathbf{A})\mathbf{D}^1 = \mathbf{0}^{n\mathrm{x}\tau} \tag{28}$$

The expression inside the square brackets is the field of residuals $\mathbf{R}_2^0$ from Equation (26). Therefore, rearrangement of Equation (28) gives

$$-\mathbf{K}\mathbf{D}^1 = \mathbf{R}_2^0 \tag{29}$$

where $\mathbf{K} = \mathbf{M} + \mathbf{A}^{\mathrm{T}}\mathbf{C}\mathbf{A}$ is the n x n diffusion operator. In the final step of the first iteration, $\mathbf{D}^1$ is to be extracted by solution of the relatively large SLE of size n x τ in (29). The next subsection will present a ROM technique to replace this SLE with a lower-dimensional one.

2.4.3 Reduced Basis and Temperature Correction

Singular Value Decomposition (SVD) is an indispensable tool in linear algebra, in large part because the parent matrix is decomposed into its four fundamental subspaces: the column space, row space, null space, and left null space. To realize the sought-after ROM for Equation (29), the SVD of $\mathbf{R}_2^0$ will be analyzed below (the 0 superscript will be suppressed for brevity).

Consider a discretization of spacetime where the number of nodes exceeds the number of time-steps, such that $\text{rank}(\mathbf{R}_2) = r \leq \tau < n$. This scenario is common in industrial-scale problems, where n is typically two to three orders of magnitude greater than τ. The SVD of $\mathbf{R}_2$ yields

$$\mathbf{R}_2 = \mathbf{H}\boldsymbol{\Sigma}\mathbf{V}^{\mathrm{T}} \tag{30}$$

where:

- $\mathbf{H} = [\mathbf{h}_1 \ \cdots \ \mathbf{h}_n]$ is an orthonormal n x n matrix having as its columns the left singular vectors $\mathbf{h}_{i=1,...,n}$ of $\mathbf{R}_2$. Since $\mathbf{H}$ has dimensions equal to the number of nodes, it is space-like and represents a basis for the spatial part of $\mathbf{R}_2$.

- $\boldsymbol{\Sigma}$ is an n x τ matrix containing the singular values $\sigma_{k=1,...,\tau}$ of $\mathbf{R}_2$ on its main diagonal. The singular values are arranged such that $\sigma_1 > \cdots > \sigma_r$, with $\sigma_k = 0$ for all $k = r+1, ..., \tau$. All entries not on the main diagonal are zero.

- $\mathbf{V} = [\mathbf{v}_1 \ \cdots \ \mathbf{v}_\tau]$ is an orthonormal τ x τ matrix having as its columns the right singular vectors $\mathbf{v}_{k=1,...,\tau}$ of $\mathbf{R}_2$. Since $\mathbf{V}$ has dimensions equal to the number of time-steps, it is time-like and represents a basis for the temporal part of $\mathbf{R}_2$.

Orthonormality of $\mathbf{H}$, for example, has the unique properties of $\mathbf{H}^{-1} = \mathbf{H}^{\mathrm{T}}$ and $\mathbf{H}\mathbf{H}^{\mathrm{T}} = \mathbf{H}^{\mathrm{T}}\mathbf{H} = \mathbf{I}^n$.

Taking inspiration from Strang [8], the row space $\mathrm{im}\left(\mathbf{R}_2^{\mathrm{T}}\right)$ and null space $\ker(\mathbf{R}_2)$ are formed by right-multiplication of both sides of Equation (30) by $\mathbf{V}$

$$\mathbf{R}_2\mathbf{v}_k = \begin{cases} \sigma_k\mathbf{h}_k & k = 1, ..., r \\ \mathbf{0}^n & k = r+1, ..., \tau \end{cases} \tag{31}$$

That is, $\mathbf{v}_{k=1,...,r}$ forms a basis for the row space of $\mathbf{R}_2$ and $\mathbf{v}_{k=r+1,...,\tau}$ the null space. Likewise, the column space $\mathrm{im}(\mathbf{R}_2)$ and left null space $\ker\left(\mathbf{R}_2^{\mathrm{T}}\right)$ are formed by left-multiplication of both sides of Equation (30) by $\mathbf{H}^{\mathrm{T}}$ (followed by transposition of both sides)

$$\mathbf{R}_2^{\mathrm{T}}\mathbf{h}_k = \begin{cases} \sigma_k\mathbf{v}_k & k = 1, ..., r \\ \mathbf{0}^\tau & k = r+1, ..., n \end{cases} \tag{32}$$

That is, $\mathbf{h}_{k=1,...,r}$ forms a basis for the column space of $\mathbf{R}_2$ and $\mathbf{h}_{k=r+1,...,n}$ the left null space. Equations (31) and (32) imply that the null space vectors $\mathbf{v}_{k=r+1,...,\tau}$ and left null space vectors $\mathbf{h}_{k=r+1,...,n}$ are extraneous to the decomposition of $\mathbf{R}_2$ and need not be formed. This means that the $\mathbf{H}$ matrix is relegated in size from n x n to n x r, which entails a significant reduction in computing and storage costs in the common scenario of $\tau \ll n$.

Building upon the theme of the previous paragraph, (30) may be recast as a sum of pairwise tensor products of the first r left and right singular vectors. Each of these tensor products is weighted by the corresponding singular value, like so

$$\mathbf{R}_2 = \sum_{k=1}^{r} \sigma_k\mathbf{h}_k \otimes \mathbf{v}_k \tag{33}$$

The dimensionality of $\mathbf{R}_2$ may be reduced by truncating the smaller magnitude terms of the summation above. This truncation is manifested as a *reduced basis* of $\mathbf{R}_2$, wherein only the $K \leq r$ largest singular values, and their corresponding left/right

singular vectors, are retained and the remaining $r - K$ values are discarded. $\mathbf{R}_2$ is replaced by the reduced model, $\widetilde{\mathbf{R}}_2 = \widetilde{\mathbf{H}}\widetilde{\mathbf{\Sigma}}\widetilde{\mathbf{V}}^{\mathrm{T}}$, where $\widetilde{\mathbf{H}},\ \widetilde{\mathbf{\Sigma}},$ and $\widetilde{\mathbf{V}}$ are of respective sizes n x $K,\ K$ x $K,$ and τ x K.

The reduced basis gives rise to an error $\Delta R \in \mathbf{R}^+$, which may be quantified using the Frobenius matrix norm symbolized by $||\cdot||_F$

$$\Delta R = ||\mathbf{R}_2 - \widetilde{\mathbf{R}}_2||_F = \sum_{i=K+1}^{r} \sigma_i^2 \tag{34}$$

In words, the error arising from the reduced basis is the sum of the squares of the $r - K$ discarded singular values. Therefore, selection of K requires a careful balance between the desired decrease in computational expense and the resultant error.

Returning now to SLE (29), the reduced basis is incorporated with the following three steps:

- $\mathbf{R}_2^0$ is replaced with $\widetilde{\mathbf{R}}_2 = \widetilde{\mathbf{H}}\widetilde{\mathbf{\Sigma}}\widetilde{\mathbf{V}}^{\mathrm{T}}$.
- Both sides of (29) are left-multiplied by $\widetilde{\mathbf{H}}^{\mathrm{T}}$.
- $\mathbf{D}^1$ is left-multiplied by $(\widetilde{\mathbf{H}}\widetilde{\mathbf{H}}^{\mathrm{T}})$.

which ultimately gives the following

$$-\overbrace{\widetilde{\mathbf{H}}^{\mathrm{T}}\mathbf{K}(\widetilde{\mathbf{H}}}^{=\widetilde{\mathbf{K}}}\overbrace{\widetilde{\mathbf{H}}^{\mathrm{T}})\mathbf{D}^1}^{=\widetilde{\mathbf{D}}^1} = \widetilde{\mathbf{H}}^{\mathrm{T}}\widetilde{\mathbf{H}}\widetilde{\mathbf{\Sigma}}\widetilde{\mathbf{V}}^{\mathrm{T}} \tag{35}$$

where $\widetilde{\mathbf{D}}^1 = \widetilde{\mathbf{H}}^{\mathrm{T}}\mathbf{D}^1$ is of size K x τ and $\widetilde{\mathbf{K}} = \widetilde{\mathbf{H}}^{\mathrm{T}}\mathbf{K}\widetilde{\mathbf{H}}$ is of size K x K and represents the reduced order diffusion operator. Orthonormality of $\widetilde{\mathbf{H}}$ simplifies the right-hand side to $\widetilde{\mathbf{\Sigma}}\widetilde{\mathbf{V}}^{\mathrm{T}}$, which gives the following concise SLE

$$-\widetilde{\mathbf{K}}\widetilde{\mathbf{D}}^1 = \widetilde{\mathbf{\Sigma}}\widetilde{\mathbf{V}}^{\mathrm{T}} \tag{36}$$

Mathematically, (35) and (36) have the effect of projecting (29) on to the reduced time-like basis. In other words, the SLE of size n x τ in (29) is replaced with the reduced SLE of size K x τ in (36).

After (36) has been solved for $\widetilde{\mathbf{D}}^1$, the temperature correction field may be computed by solution of the SLE $\widetilde{\mathbf{D}}^1 = \widetilde{\mathbf{H}}^{\mathrm{T}}\mathbf{D}^1$. Orthonormality of $\widetilde{\mathbf{H}}$ simplifies this final SLE to $\mathbf{D}^1 = \widetilde{\mathbf{H}}\widetilde{\mathbf{D}}^1$. The correction is then applied to the trial temperature field per $\widehat{\mathbf{U}}^1 = \widehat{\mathbf{U}}^0 + \mathbf{D}^1$.

Having now come full circle, $\widehat{\mathbf{U}}^1$ is used to initialize the next LATIN iteration. The algorithm iterates until the solution converges to within some tolerance of the exact discrete solution $(\mathbf{Q}^*, \mathbf{U}^*)$. Section 2.5 will discuss a duality-derived error bound and how it may be used as a convergence criterion.

The LATIN algorithm is summarized below in ten steps:

1. The user inputs a KA dual temperature field $\widehat{\mathbf{U}}^0 \in \mathscr{K}_D$. $\widehat{\mathbf{U}}^0$ must satisfy the initial temperature field $u_0(x)$.

2. The KA primal flux field $\widehat{\mathbf{Q}}^0 \in \mathscr{K}_P$ is computed from Fourier's law and $\widehat{\mathbf{U}}^0$.

3. $\mathbf{R}_2^0$ measures the non-conformity of the trial pair $(\widehat{\mathbf{Q}}^0, \widehat{\mathbf{U}}^0)$ with the heat equation.

4. $\widehat{\mathbf{Q}}^0$ is interpolated from the m line segments to the N nodes. The output, labelled $\widehat{\mathbf{Q}}^{1/2}$, will not generally satisfy the Neumann BC $\widetilde{q}(t)$.

5. $\widehat{\mathbf{Q}}^{1/2}$ is then interpolated back on to the m line segments while enforcing the Neumann BC. This second interpolation gives the SA flux field $\overline{\mathbf{Q}}^{1/2}$.

6. A ROM for SLE (29) is favoured over direct solution. The first step of the order reduction performs SVD on the $\mathbf{R}_2^0$ matrix.

7. The $\tau - K$ smallest singular values and associated left and right singular vectors of $\mathbf{R}_2^0$ are discarded.

8. SLE (29) is projected on to the reduced time-like basis and then solved for the field of temperature corrections, $\mathbf{D}^1$.

9. The initial temperature field is corrected as $\widehat{\mathbf{U}}^1 = \widehat{\mathbf{U}}^0 + \mathbf{D}^1$.

10. Steps two through nine are repeated using $\widehat{\mathbf{U}}^1$ as the trial solution. The algorithm iterates until the solution converges to within some user-defined tolerance of $(\mathbf{Q}^*, \mathbf{U}^*)$.

2.5 LATIN Error Bound

The power of the duality concept lies in its realization as an error bound for the LATIN algorithm. As such, this section will identify the link between the duality gap $\widetilde{d}$ (the difference in primal and dual energies) and the residual field $\mathbf{R}_1$ (a measure of the non-conformity of a given flux-temperature pair with Fourier's law).

Consider the discrete Lagrangian function, as stated in Equation (21) and repeated here for ease of reference, for an arbitrarily selected pair of flux-temperature fields, $(\mathbf{Q}, \mathbf{U})$. For the moment, no restrictions are placed upon $\mathbf{Q}$ and $\mathbf{U}$ with respect to kinematic or static admissibility

$$\mathcal{L}(\mathbf{Q}, \mathbf{U}) = \mathrm{tr}\left[\frac{1}{2}\mathbf{Q}^{\mathrm{T}}\mathbf{C}^{-1}\mathbf{Q} - \mathbf{F}_1^{\mathrm{T}}\mathbf{Q} - \mathbf{U}^{\mathrm{T}}(-\mathbf{A}^{\mathrm{T}}\mathbf{Q} + \mathbf{M}\mathbf{U} - \mathbf{F}_2)\right]\Delta t$$

Begin by deriving the expressions for the primal and dual energies. $P(\mathbf{Q})$ is derived by first maximizing the Lagrangian over all temperature fields, similarly to the formulation of Problem L.1 in Section 2.1. Enforcement of $\partial\mathcal{L}/\partial u_{i,k} = 0$ over $i = \widetilde{n} + 1, ..., N, k = 1, ..., \tau$ culminates in a SLE of size n x τ

$$-\mathbf{A}^{\mathrm{T}}\mathbf{Q} + \mathbf{M}\mathbf{U} = \mathbf{F}_2 \tag{37}$$

which may be recognized as the discrete form of the heat equation. Therefore, the primal energy follows by substitution of a SA pair into the Lagrangian function, such that $\mathbf{Q} \Rightarrow \overline{\mathbf{Q}}$ and $\mathbf{U} \Rightarrow \overline{\mathbf{U}} = \mathbf{M}^{-1}(\mathbf{F}_2 + \mathbf{A}^{\mathrm{T}}\overline{\mathbf{Q}})$

$$P(\overline{\mathbf{Q}}) = \mathcal{L}(\overline{\mathbf{Q}}, \overline{\mathbf{U}}) = \mathrm{tr}\left(\frac{1}{2}\overline{\mathbf{Q}}^{\mathrm{T}}\mathbf{C}^{-1}\overline{\mathbf{Q}} - \mathbf{F}_1^{\mathrm{T}}\overline{\mathbf{Q}}\right)\Delta t \tag{38}$$

$D(\mathbf{U})$ is derived by first minimizing the Lagrangian over all flux fields, similarly to the formulation of Problem L.2 in Section 2.1. Enforcement of $\partial\mathcal{L}/\partial\mathbf{q}_{i,k} = 0$ over $i = 1, ..., m, k = 1, ..., \tau$ culminates in a SLE of size m x τ

$$\mathbf{C}^{-1}\mathbf{Q} + \mathbf{A}\mathbf{U} = \mathbf{F}_1 \tag{39}$$

which may be recognized as the discrete form of Fourier's law. Therefore, the dual energy follows by substitution of a KA pair into the Lagrangian function, such that $\mathbf{U} \Rightarrow \widehat{\mathbf{U}}$ and $\mathbf{Q} \Rightarrow \widehat{\mathbf{Q}} = \mathbf{C}(\mathbf{F}_1 - \mathbf{A}\widehat{\mathbf{U}})$

$$\begin{aligned} D(\widehat{\mathbf{U}}) = \mathcal{L}(\widehat{\mathbf{Q}}, \widehat{\mathbf{U}}) = \mathrm{tr}\Bigg[\frac{1}{2}(\mathbf{F}_1 - \mathbf{A}\widehat{\mathbf{U}})^{\mathrm{T}}\mathbf{C}(\mathbf{F}_1 - \mathbf{A}\widehat{\mathbf{U}}) + \\ \Bigg\{ - \mathbf{F}_1^{\mathrm{T}}\mathbf{C}(\mathbf{F}_1 - \mathbf{A}\widehat{\mathbf{U}}) - \widehat{\mathbf{U}}^{\mathrm{T}}\Big(- \mathbf{A}^{\mathrm{T}}\mathbf{C}(\mathbf{F}_1 - \mathbf{A}\widehat{\mathbf{U}}) + \mathbf{M}\widehat{\mathbf{U}} - \mathbf{F}_2\Big)\Bigg\}\Bigg]\Delta t \end{aligned} \tag{40}$$

To streamline SLE (40), the term in curled braces is considered in isolation for the moment. Rearrangement of the enclosed expression gives

$$\Big(- \mathbf{F}_1^{\mathrm{T}}\mathbf{C}(\mathbf{F}_1 - \mathbf{A}\widehat{\mathbf{U}}) + \widehat{\mathbf{U}}^{\mathrm{T}}\mathbf{A}^{\mathrm{T}}\mathbf{C}(\mathbf{F}_1 - \mathbf{A}\widehat{\mathbf{U}})\Big) - \widehat{\mathbf{U}}^{\mathrm{T}}(\mathbf{M}\widehat{\mathbf{U}} - \mathbf{F}_2) \tag{41}$$

The term inside the large parentheses may in turn be factored to produce

$$(-\mathbf{F}_1^{\mathrm{T}} + \widehat{\mathbf{U}}^{\mathrm{T}}\mathbf{A}^{\mathrm{T}})\mathbf{C}(\mathbf{F}_1 - \mathbf{A}\widehat{\mathbf{U}}) - \widehat{\mathbf{U}}^{\mathrm{T}}(\mathbf{M}\widehat{\mathbf{U}} - \mathbf{F}_2) \tag{42}$$

Simplification of expression (42) arises with the following sequence of replacements: $(-\mathbf{F}_1^{\mathrm{T}} + \widehat{\mathbf{U}}^{\mathrm{T}}\mathbf{A}^{\mathrm{T}}) \Rightarrow -(\mathbf{F}_1^{\mathrm{T}} - \widehat{\mathbf{U}}^{\mathrm{T}}\mathbf{A}^{\mathrm{T}}) \Rightarrow -(\mathbf{F}_1 - \mathbf{A}\widehat{\mathbf{U}})^{\mathrm{T}}$, where the latter replacement follows from the properties of matrix transposition. These substitutions give the following expression

$$-(\mathbf{F}_1 - \mathbf{A}\widehat{\mathbf{U}})^{\mathrm{T}}\mathbf{C}(\mathbf{F}_1 - \mathbf{A}\widehat{\mathbf{U}}) - \widehat{\mathbf{U}}^{\mathrm{T}}(\mathbf{M}\widehat{\mathbf{U}} - \mathbf{F}_2) \tag{43}$$

Expression (43) may now be reintroduced into (40) to yield a simplified $D(\widehat{\mathbf{U}})$

$$D(\widehat{\mathbf{U}}) = \mathrm{tr}\left[\frac{1}{2}(\mathbf{F}_1 - \mathbf{A}\widehat{\mathbf{U}})^{\mathrm{T}}\mathbf{C}(\mathbf{F}_1 - \mathbf{A}\widehat{\mathbf{U}}) - (\mathbf{F}_1 - \mathbf{A}\widehat{\mathbf{U}})^{\mathrm{T}}\mathbf{C}(\mathbf{F}_1 - \mathbf{A}\widehat{\mathbf{U}}) - \widehat{\mathbf{U}}^{\mathrm{T}}(\mathbf{M}\widehat{\mathbf{U}} - \mathbf{F}_2)\right]\Delta t \tag{44}$$

which in turn reduces to

$$D(\widehat{\mathbf{U}}) = \mathrm{tr}\left[-\frac{1}{2}(\mathbf{F}_1 - \mathbf{A}\widehat{\mathbf{U}})^{\mathrm{T}}\mathbf{C}(\mathbf{F}_1 - \mathbf{A}\widehat{\mathbf{U}}) - \widehat{\mathbf{U}}^{\mathrm{T}}(\mathbf{M}\widehat{\mathbf{U}} - \mathbf{F}_2)\right]\Delta t \tag{45}$$

Once again making use of the linearity of the trace operation, the duality gap results from subtraction of Equation (45) from (38)

$$\widetilde{d} = P(\overline{\mathbf{Q}}) - D(\widehat{\mathbf{U}}) =$$
$$\mathrm{tr}\left[\left(\frac{1}{2}\overline{\mathbf{Q}}^{\mathrm{T}}\mathbf{C}^{-1}\overline{\mathbf{Q}} - \mathbf{F}_1^{\mathrm{T}}\overline{\mathbf{Q}}\right) - \left(-\frac{1}{2}(\mathbf{F}_1 - \mathbf{A}\widehat{\mathbf{U}})^{\mathrm{T}}\mathbf{C}(\mathbf{F}_1 - \mathbf{A}\widehat{\mathbf{U}}) - \widehat{\mathbf{U}}^{\mathrm{T}}(\mathbf{M}\widehat{\mathbf{U}} - \mathbf{F}_2)\right)\right]\Delta t \tag{46}$$

Equation (37) permits the substitution $\widehat{\mathbf{U}}^{\mathrm{T}}(\mathbf{M}\widehat{\mathbf{U}} - \mathbf{F}_2) \Rightarrow \widehat{\mathbf{U}}^{\mathrm{T}}(\mathbf{A}^{\mathrm{T}}\overline{\mathbf{Q}})$ and combination of this result with the $\mathbf{F}_1^{\mathrm{T}}\overline{\mathbf{Q}}$ term above yields $(-\mathbf{F}_1^{\mathrm{T}} + \widehat{\mathbf{U}}^{\mathrm{T}}\mathbf{A}^{\mathrm{T}})\overline{\mathbf{Q}}$. Making use of the same sequence of replacements as above gives the equation below

$$\widetilde{d} = \mathrm{tr}\left[\frac{1}{2}\overline{\mathbf{Q}}^{\mathrm{T}}\mathbf{C}^{-1}\overline{\mathbf{Q}} - (\mathbf{F}_1 - \mathbf{A}\widehat{\mathbf{U}})^{\mathrm{T}}\overline{\mathbf{Q}} + \frac{1}{2}(\mathbf{F}_1 - \mathbf{A}\widehat{\mathbf{U}})^{\mathrm{T}}\mathbf{C}(\mathbf{F}_1 - \mathbf{A}\widehat{\mathbf{U}})\right]\Delta t \tag{47}$$

Equation (47) suggests that the argument of the trace (enclosed in square brackets above) is a quadratic form in the variables $\overline{\mathbf{Q}}$ and $\widehat{\mathbf{U}}$; it may be factored like so

$$\widetilde{d} = \mathrm{tr}\left[\frac{1}{2}(\mathbf{C}^{-1}\overline{\mathbf{Q}} + \mathbf{A}\widehat{\mathbf{U}} - \mathbf{F}_1)^{\mathrm{T}}\mathbf{C}(\mathbf{C}^{-1}\overline{\mathbf{Q}} + \mathbf{A}\widehat{\mathbf{U}} - \mathbf{F}_1)\right]\Delta t \tag{48}$$

Upon comparison of Equations (48) and (24), the two terms above in parentheses may be recognized as the spacetime field of residuals in Fourier's law, or $\mathbf{R}_1$. This insight permits a more concise expression for the duality gap

$$\widetilde{d} = \frac{\Delta t}{2}\mathrm{tr}(\mathbf{R}_1^{\mathrm{T}}\mathbf{C}\mathbf{R}_1) \tag{49}$$

Via weak duality, the scalar $\widetilde{d}$ returned by Equation (49) bounds the error at each iteration of the LATIN algorithm. This concept is best illustrated by the introduction of two scalar error values, $\delta_P^p, \delta_D^p \in \mathbb{R}^+$, which measure the respective errors in the primal and dual energies for the p^{th} LATIN iteration

$$\begin{aligned} \delta_P^p &= P(\overline{\mathbf{Q}}^p) - \mathcal{L}(\mathbf{Q}^*, \mathbf{U}^*) \\ \delta_D^p &= \mathcal{L}(\mathbf{Q}^*, \mathbf{U}^*) - D(\widehat{\mathbf{U}}^p) \end{aligned} \tag{50}$$

where $\mathcal{L}(\mathbf{Q}^*, \mathbf{U}^*)$ represents the energy of the exact discrete solution. Addition of the two equations above gives $\delta_P^p + \delta_D^p = \widetilde{d}^p$ and, since the primal and dual errors are strictly positive real numbers, $\delta_P^p, \delta_D^p < \widetilde{d}^p$ follows logically.

To summarize, the KA $\widehat{\mathbf{U}}$ from step one and SA $\overline{\mathbf{Q}}$ from step five of the LATIN algorithm on page 50 enable calculation of the field of Fourier's law residuals, $\mathbf{R}_1$. Via weak duality, $\widetilde{d} = \mathrm{tr}(\mathbf{R}_1^{\mathrm{T}}\mathbf{C}\mathbf{R}_1)\Delta t/2$ bounds the exact energy $\mathcal{L}(\mathbf{Q}^*, \mathbf{U}^*)$ between $P(\overline{\mathbf{Q}})$ from above and $D(\widehat{\mathbf{U}})$ from below. Therefore, $\widetilde{d}$ provides a simple convergence criterion for the LATIN algorithm with $\overline{\mathbf{Q}} \to \mathbf{Q}^*$ and $\widehat{\mathbf{U}} \to \mathbf{U}^*$ as $\widetilde{d} \to 0$.

To conclude this chapter, a brief discussion of the convexity of P is in order. As

discussed in Section 2.2, the convexity of P is governed by the positive or semi-positive definiteness of its Hessian matrix. By inspection of Equation (38), $\mathbf{H}(P) = (\mathbf{C}^{-1})^{\mathrm{T}} = \mathbf{C}^{-1}$ by the symmetry of the $\mathbf{C}$ matrix, or

$$\begin{bmatrix} \frac{\partial^2 P}{\partial \mathbf{q}_1^2} & \cdots & \frac{\partial^2 P}{\partial \mathbf{q}_1 \partial \mathbf{q}_i} & \cdots & \frac{\partial^2 P}{\partial \mathbf{q}_1 \partial \mathbf{q}_m} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \frac{\partial^2 P}{\partial \mathbf{q}_i \partial \mathbf{q}_1} & \cdots & \frac{\partial^2 P}{\partial \mathbf{q}_i^2} & \cdots & \frac{\partial^2 P}{\partial \mathbf{q}_i \partial \mathbf{q}_m} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \frac{\partial^2 P}{\partial \mathbf{q}_m \partial \mathbf{q}_1} & \cdots & \frac{\partial^2 P}{\partial \mathbf{q}_m \partial \mathbf{q}_i} & \cdots & \frac{\partial^2 P}{\partial \mathbf{q}_m^2} \end{bmatrix} = \kappa^{-1} \begin{bmatrix} \langle \phi_1^q, \phi_1^q \rangle_\Omega & \cdots & \langle \phi_1^q, \phi_i^q \rangle_\Omega & \cdots & \langle \phi_1^q, \phi_m^q \rangle_\Omega \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \langle \phi_i^q, \phi_1^q \rangle_\Omega & \cdots & \langle \phi_i^q, \phi_i^q \rangle_\Omega & \cdots & \langle \phi_i^q, \phi_m^q \rangle_\Omega \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \langle \phi_m^q, \phi_1^q \rangle_\Omega & \cdots & \langle \phi_m^q, \phi_i^q \rangle_\Omega & \cdots & \langle \phi_m^q, \phi_m^q \rangle_\Omega \end{bmatrix} \tag{51}$$

Since $\mathbf{C}^{-1}$ is a Gramian matrix and solely dependent on user-input, then $\mathbf{H}(P)$ is positive or semi-positive definite for an appropriate choice of ϕ^q. Simply put, P may be chosen convex with selection of the basis functions; see Sections 4.1 and 4.2 for more details. Recall from Section 2.2 that D is concave regardless of the behaviour of P.

Chapter 3

Radial Basis Functions

Chapter 3 opens with a mathematical treatment of RBF scattered data interpolation, which is segregated into standard and Hermite formulations. The discretization of differential operators with RBFs follows, concluding with a discussion of some of the mathematical properties of the Gaussian RBF kernel.

General symbols:

Symbol	**Definition and (Units)**
$\{\cdot,\cdot\}$	indicator function
$\|\|\cdot\|\|_2$	Euclidean norm
$\mathbf{0}^z$	column vector of zeros of length z
$\mathbf{0}^{z\mathrm{x}z}$	matrix of zeros of size z x z
$\mathscr{A} \in \mathbb{R}^{M\mathrm{x}m}$	given set of scattered data centers, $\boldsymbol{\alpha}_{i=1,\ldots,M}$
$\mathscr{A}^G \in \mathbb{R}^{M_{\mathfrak{N}}\mathrm{x}m}$	set of ghost centers, $\boldsymbol{\alpha}^G_{i=1,\ldots,M_{\mathfrak{N}}}$
$\mathscr{A}^\dagger = \mathscr{A} \cup \mathscr{A}^G$	the combined set of data centers and ghost centers

$\mathscr{B} \in \mathbb{R}^{M_x \mathrm{x} m}$	given set of scattered x-derivative centers, $\boldsymbol{\beta}_{i=1,\ldots,M_x}$
$\mathbf{B}(\mathscr{K}[\mathbf{r}])$	collocation matrix for the stencil $\mathscr{K}[\mathbf{r}]$
C^p	the set of all functions which have $p \in \mathbb{N}$ continuous derivatives
$d \in \mathbb{N}$	polynomial degree
$d^\phi(\mathbf{r}, \boldsymbol{\alpha}) :$ Ω x $\mathscr{A} \mapsto \mathbb{R}$	discrete form of ∇ which operates locally on $u(\boldsymbol{\alpha}),\ \boldsymbol{\alpha} \in \mathscr{A}$
$\mathbf{d}^\phi(\mathbf{r}, \mathscr{K}[\boldsymbol{\alpha}]) \in \mathbb{R}^k$	row vector with entries $d^\phi(\mathbf{r}, \boldsymbol{\alpha}),\ \forall \boldsymbol{\alpha} \in \mathscr{K}[\boldsymbol{\alpha}]$
$\mathscr{F}$	field of data $f_{i=1,\ldots,M}$ given on $\mathscr{A}$
$\mathscr{F}_x$	field of x-derivative data $\partial f_{i=1,\ldots,M_x}/\partial x$ given on $\mathscr{B}$
$\mathscr{F}_y$	field of y-derivative data $\partial f_{i=1,\ldots,M_y}/\partial y$ given on $\mathscr{G}$
$\mathscr{F}[\mathbf{r}] \subseteq \mathscr{F}$	data subset located on $\mathscr{K}[\mathbf{r}] \subseteq \mathscr{A}$
$\mathscr{F}_x[\mathbf{r}] \subseteq \mathscr{F}_x$	x-derivative data subset located on $\mathscr{K}_x[\mathbf{r}] \subseteq \mathscr{B}$
$\mathscr{F}_y[\mathbf{r}] \subseteq \mathscr{F}_y$	y-derivative data subset located on $\mathscr{K}_y[\mathbf{r}] \subseteq \mathscr{G}$
$f(\mathbf{r}) : \Omega \mapsto \mathbb{R}$	prescribed scalar function on Ω
$\mathbf{f}(\mathscr{A}) \in \mathbb{R}^M$	discrete global field consisting of $f(\mathbf{r})\Big\vert_{\mathbf{r}=\boldsymbol{\alpha}},\ \forall \boldsymbol{\alpha} \in \mathscr{A}$
$\mathbf{f}(\mathscr{A}^\dagger) \in \mathbb{R}^{M+M_N}$	discrete global field consisting of $f(\mathbf{r})\Big\vert_{\mathbf{r}=\boldsymbol{\alpha}},\ \forall \boldsymbol{\alpha} \in \mathscr{A}^\dagger$
$\mathscr{G} \in \mathbb{R}^{M_y \mathrm{x} m}$	given set of scattered y-derivative centers, $\boldsymbol{\gamma}_{i=1,\ldots,M_y}$
$g(\mathbf{r}) : \Gamma_D \mapsto \mathbb{R}$	prescribed scalar Dirichlet boundary function on Γ_D
$h(\mathbf{r}) : \Gamma_N \mapsto \mathbb{R}$	prescribed scalar Neumann boundary function on Γ_N

$\mathscr{K}[\mathbf{r}] \subseteq \mathscr{A}$	the k data centers nearest to an arbitrary location $\mathbf{r}$
$\mathscr{K}_x[\mathbf{r}] \subseteq \mathscr{B}$	the k_x x-derivative centers nearest to an arbitrary location $\mathbf{r}$
$\mathscr{K}_y[\mathbf{r}] \subseteq \mathscr{G}$	the k_y y-derivative centers nearest to an arbitrary location $\mathbf{r}$
$k \leq M$	quantity of data centers used in $\mathscr{K}[\mathbf{r}]$
$k_x \leq M_x$	quantity of x-derivative centers used in $\mathscr{K}_x[\mathbf{r}]$
$k_y \leq M_y$	quantity of y-derivative centers used in $\mathscr{K}_y[\mathbf{r}]$
$\mathscr{L}$	continuous, linear differential operator
$\ell^\phi(\mathbf{r}, \boldsymbol{\alpha})$: Ω x $\mathscr{A} \mapsto \mathbb{R}$	discrete form of $\mathscr{L}$ which operates locally on an arbitrary $u(\boldsymbol{\alpha})$, $\boldsymbol{\alpha} \in \mathscr{A}$
$\boldsymbol{\ell}^\phi(\mathbf{r}, \mathscr{K}[\mathbf{r}]) \in \mathbb{R}^k$	row vector with entries $\ell^\phi(\mathbf{r}, \boldsymbol{\alpha}),\ \forall \boldsymbol{\alpha} \in \mathscr{K}[\mathbf{r}]$
$\mathbf{L} \in \mathbb{R}^{M \text{x} M}$	discrete form of $\mathscr{L}$ which operates globally on $u(\boldsymbol{\alpha}),\ \forall \boldsymbol{\alpha} \in \mathscr{A}$
$m \in \mathbb{N}$	dimension of the Euclidean space containing all data and derivative centers
$M \in \mathbb{N}$	quantity of data centers
$M_x \in \mathbb{N}$	quantity of x-derivative centers
$M_y \in \mathbb{N}$	quantity of y-derivative centers
$M_{\mathfrak{N}} \in \mathbb{N}$	quantity of data centers on $\Gamma_{\mathfrak{N}}$
$p_i(\mathbf{r}) : \mathbb{R}^m \mapsto \mathbb{R}$	the i^{th} monomial of the degree d polynomial
$\mathbf{r} \in \mathbb{R}^m$	position vector

$s(\mathbf{r})$	function which interpolates $\mathscr{F}$ to an arbitrary location $\mathbf{r}$. The output of $s(\mathbf{r})$ has the same format as the data in $\mathscr{F}$
$u(\boldsymbol{\alpha}) \in \mathbb{R}$	unknown value of $u(\mathbf{r})$ at any arbitrary data center $\boldsymbol{\alpha} \in \mathscr{A}$
$u(\mathbf{r}) : \Omega \cup \Gamma \mapsto \mathbb{R}$	unknown scalar function on $\Omega \cup \Gamma$
$\mathbf{u}(\mathscr{K}[\mathbf{r}]) \in \mathbb{R}^k$	discrete local field consisting of $u(\boldsymbol{\alpha}),\ \forall \boldsymbol{\alpha} \in \mathscr{K}[\mathbf{r}]$
$\mathbf{u}(\mathscr{A}) \in \mathbb{R}^M$	discrete global field consisting of $u(\boldsymbol{\alpha}),\ \forall \boldsymbol{\alpha} \in \mathscr{A}$
$\mathbf{u}(\mathscr{A}^\dagger) \in \mathbb{R}^{M+M_\mathfrak{N}}$	discrete global field consisting of $u(\boldsymbol{\alpha}),\ \forall \boldsymbol{\alpha} \in \mathscr{A}^\dagger$
$w(\boldsymbol{\alpha}) \in \mathbb{R}$	scalar weight factor of the RBK, $\phi(\mathbf{r}, \boldsymbol{\alpha})$
$w_x(\boldsymbol{\beta}) \in \mathbb{R}$	scalar weight factor of the RBK derivative, $\partial\phi(\mathbf{r}, \boldsymbol{\beta})/\partial x$
$w_y(\boldsymbol{\gamma}) \in \mathbb{R}$	scalar weight factor of the RBK derivative, $\partial\phi(\mathbf{r}, \boldsymbol{\gamma})/\partial y$
$\mathbf{w} \in \mathbb{R}^k$	vector containing the weight factors $w(\boldsymbol{\alpha}),\ \forall \boldsymbol{\alpha} \in \mathscr{K}[\mathbf{r}]$
$\mathbf{w}_x \in \mathbb{R}^{k_x}$	vector containing the weight factors $w_x(\boldsymbol{\beta}),\ \forall \boldsymbol{\beta} \in \mathscr{K}_x[\mathbf{r}]$
$\mathbf{w}_y \in \mathbb{R}^{k_y}$	vector containing the weight factors $w_y(\boldsymbol{\gamma}),\ \forall \boldsymbol{\gamma} \in \mathscr{K}_y[\mathbf{r}]$
$z = \binom{d+m}{d}$	number of monomials in the degree d polynomial

Greek symbols:

Symbol	**Definition and (Units)**
$\boldsymbol{\alpha} \in \mathscr{A}$	m-dimensional coordinate vector of any arbitrary data center
$\boldsymbol{\alpha}^G \in \mathscr{A}^G$	m-dimensional coordinate vector of any arbitrary ghost center

$\boldsymbol{\beta} \in \mathscr{B}$	m-dimensional coordinate vector of any arbitrary x-derivative center
$\delta_{i,j}$	Kronecker delta with respect to indices i and j
$\epsilon \in \mathbb{R}^+$	RBF shape factor
$\boldsymbol{\gamma} \in \mathscr{G}$	m-dimensional coordinate vector of any arbitrary y-derivative center
$\Gamma = \partial\Omega$	closed boundary of Ω
$\Gamma_D \subseteq \Gamma$	sub-boundary where the Dirichlet BCs are prescribed
$\Gamma_N \subseteq \Gamma$	sub-boundary where the Neumann BCs are prescribed
$\lambda_i \in \mathbb{R}$	scalar coefficient of the i^{th} monomial
$\boldsymbol{\lambda} \in \mathbb{R}^z$	vector containing the monomial coefficients $\lambda_{i=1,\ldots,z}$
$\nabla\square$	gradient operator on the field or vector $\square$
$\Omega \subseteq \mathbb{R}^3$	three-dimensional domain
$\phi(\mathbf{r}) : \mathbb{R}^m \mapsto \mathbb{R}$	scalar RBF
$\phi(\mathbf{r}, \boldsymbol{\alpha}) :$ $\mathbb{R}^m$ x $\mathscr{A} \mapsto \mathbb{R}$	the RBK evaluated at $\mathbf{r}$ and centered at $\boldsymbol{\alpha}$
$\dfrac{\partial\phi(\mathbf{r}, \boldsymbol{\beta})}{\partial x} :$ $\mathbb{R}^m$ x $\mathscr{B} \mapsto \mathbb{R}$	x-derivative of the RBK evaluated at $\mathbf{r}$ and centered at $\boldsymbol{\beta}$
$\dfrac{\partial\phi(\mathbf{r}, \boldsymbol{\gamma})}{\partial y} :$ $\mathbb{R}^m$ x $\mathscr{G} \mapsto \mathbb{R}$	y-derivative of the RBK evaluated at $\mathbf{r}$ and centered at $\boldsymbol{\gamma}$

3.1 Scattered Data Interpolation

3.1.1 Standard Interpolation

A set of scattered test coordinates $\mathscr{A} = [\boldsymbol{\alpha}_1 \ \cdots \ \boldsymbol{\alpha}_i \ \cdots \ \boldsymbol{\alpha}_M]^{\mathrm{T}}$ is given, where each of the $\boldsymbol{\alpha}$ lie in m-dimensional Euclidean space $\mathbb{R}^m$. $\mathscr{A}$ will henceforth be referred to as the set of *data centers*. A data field $\mathscr{F} = [f_1 \ \cdots \ f_i \ \cdots \ f_M]^{\mathrm{T}}$ is also given, where each f is associated with a data center. In general, $\mathscr{F}$ may vary in complexity from scalar fields (e.g. temperature) to tensor fields (e.g. Cauchy stress). The objective is to determine a function $s(\mathbf{r})$, called an interpolant, which interpolates $\mathscr{F}$ to any point $\mathbf{r} \in \mathbb{R}^m$ not contained in $\mathscr{A}$.

An increasingly popular method of scattered data interpolation formulates a basis for $s(\mathbf{r})$ by placing an RBF at each of the M data centers. The RBFs may take many forms, with popular choices including Gaussian, multiquadric [35], and thin plate spline [36]. These basis functions all share the characteristics of translational and rotational invariance [18].

Related to the RBF concept is the RBK; formally, the RBK is defined by the mapping $\phi(\mathbf{r}, \boldsymbol{\alpha}) : \mathbb{R}^m \text{ x } \mathscr{A} \rightarrow \mathbb{R}$. Informally, the RBK takes the coordinates of $\mathbf{r}$ and $\boldsymbol{\alpha}$ as input and outputs the "correlation" between the pair which is generally proportional to their proximity in space. The form of ϕ will be left general for now and further discussion of its mathematical properties will be left to Section 3.3.

Consider a subset of the M data centers containing only the k centers nearest to $\mathbf{r}$. This subset $\mathscr{K}[\mathbf{r}] \subseteq \mathscr{A}$ is also referred to as the k-stencil of $\mathbf{r}$, and it may be used to segregate the RBF interpolation framework into two subclasses:

- The *globally-supported* subclass with $\mathscr{K}[\mathbf{r}] = \mathscr{A}$, in which all $k = M$ data centers and RBFs are used as a basis for $s(\mathbf{r})$.

- The *locally-supported* subclass with $\mathscr{K}[\mathbf{r}] \subset \mathscr{A}$, in which only the $k < M$ data centers and RBFs nearest to $\mathbf{r}$ are used as a basis for $s(\mathbf{r})$. By extension, $\mathscr{F}[\mathbf{r}] \subset \mathscr{F}$ symbolizes the subset of data located at the data centers in $\mathscr{K}[\mathbf{r}]$.

$s(\mathbf{r})$ is then a linear combination of RBKs with an appended polynomial of degree d (see [17], [23], and [24] for justification of the polynomial)

$$s(\mathbf{r}) = \sum_{\boldsymbol{\alpha} \in \mathscr{K}[\mathbf{r}]} w(\boldsymbol{\alpha}) \phi(\mathbf{r}, \boldsymbol{\alpha}) + \sum_{i=1}^{z} \lambda_i p_i(\mathbf{r}) \tag{52}$$

where $w(\boldsymbol{\alpha}) \in \mathbb{R}$ are the scalar weight factors for the RBKs, p_i denotes the i^{th} term of the polynomial, $\lambda_i \in \mathbb{R}$ is the coefficient of p_i, and $z = \binom{d+m}{d}$ is the binomial coefficient representing the total number of terms in the polynomial. The unisolvency theorem dictates that d be selected such that $z \leq k - 1$ in order to ensure uniqueness of the polynomial on each k-stencil.

The $\sum_{\boldsymbol{\alpha} \in \mathscr{K}[\mathbf{r}]} w(\boldsymbol{\alpha}) \phi(\mathbf{r}, \boldsymbol{\alpha})$ summation convention used in Equation (52) is equivalent to the more familiar $\sum_{i=1}^{k} w(\boldsymbol{\alpha}_i) \phi(\mathbf{r}, \boldsymbol{\alpha}_i)$ style. That is, the subscript of $\sum$ indicates summation over each of the data centers $\boldsymbol{\alpha}$ in $\mathscr{K}[\mathbf{r}]$.

To clarify the notation used above for the polynomial, $p_{i=1,\ldots,z}$ is written below for an $m = 2$-dimensional domain with $\mathbf{r} = (x, y)$

$$p_{i=1,\ldots,z}(x, y) = \tag{53}$$

$$\begin{array}{cccccccccc} x^0y^0, & x^1y^0, & x^0y^1, & \cdots & x^dy^0, & x^{d-1}y^1, & \cdots & x^1y^{d-1}, & x^0y^d \\ p_1 & p_2 & p_3 & \cdots & p_{\frac{(d+1)(d+2)+1}{2}} & p_{\frac{(d+1)(d+2)}{2}} & \cdots & p_{z-1} & p_z \end{array}$$

The RBK weights and polynomial coefficients are entered into vectors $\mathbf{w}$ and $\boldsymbol{\lambda}$ of respective lengths k and z. Two sets of constraints are required to determine $\mathbf{w}$ and $\boldsymbol{\lambda}$: the first, called collocation, arises from constraining the interpolant to equal the data at each data center in the stencil, or $\left\{ s(\mathbf{r})\Big|_{\mathbf{r}=\boldsymbol{\alpha}} = f,\ \forall \boldsymbol{\alpha} \in \mathscr{K}[\mathbf{r}],\ \forall f \in \mathscr{F}[\mathbf{r}] \right\}$. The

second set constrains the moments of the RBK weight factors and polynomial basis to vanish at each of the data centers in the k-stencil

$$\begin{aligned} \sum_{\boldsymbol{\alpha}\in\mathscr{K}[\mathbf{r}]} w(\boldsymbol{\alpha})p_1(\boldsymbol{\alpha}) = \sum_{\boldsymbol{\alpha}\in\mathscr{K}[\mathbf{r}]} w(\boldsymbol{\alpha})p_2(\boldsymbol{\alpha}) = \cdots = \\ \sum_{\boldsymbol{\alpha}\in\mathscr{K}[\mathbf{r}]} w(\boldsymbol{\alpha})p_{z-1}(\boldsymbol{\alpha}) = \sum_{\boldsymbol{\alpha}\in\mathscr{K}[\mathbf{r}]} w(\boldsymbol{\alpha})p_z(\boldsymbol{\alpha}) = 0 \end{aligned} \tag{54}$$

Mathematically, the constraint set embodied by Equation (54) has the effect of imposing orthogonality between the two function spaces containing the RBFs and polynomials. The two constraint sets defined above culminate in the following symmetric SLE of size $(k+z)$ x $(k+z)$

$$\underbrace{\begin{bmatrix} \boldsymbol{\Phi}(\mathscr{K}[\mathbf{r}],\mathscr{K}[\mathbf{r}]) & \mathbf{P}(\mathscr{K}[\mathbf{r}]) \\ \mathbf{P}(\mathscr{K}[\mathbf{r}])^{\mathrm{T}} & \mathbf{0}^{z\mathrm{x}z} \end{bmatrix}}_{=\mathbf{B}(\mathscr{K}[\mathbf{r}])} \begin{bmatrix} \mathbf{w} \\ \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} \mathscr{F}[\mathbf{r}] \\ \mathbf{0}^z \end{bmatrix} \tag{55}$$

As indicated by the underbrace above, the collocation matrix for $\mathscr{K}[\mathbf{r}]$ will be symbolized throughout the rest of this chapter by $\mathbf{B}(\mathscr{K}[\mathbf{r}])$. Appendix B includes details on the structure of sub-matrices $\boldsymbol{\Phi}$ and $\mathbf{P}$. To conclude the interpolation process, $\mathbf{B}(\mathscr{K}[\mathbf{r}])$ is naively assumed to be invertible; see Section 3.3 for a brief discussion. Consequently, solution of Equation (55) gives the vectors $\mathbf{w}$ and $\boldsymbol{\lambda}$, which may then be substituted into Equation (52) for $s(\mathbf{r})$.

For multiple interpolation centers $\mathbf{r}_1, \mathbf{r}_2, ...$ each with locally-supported stencils $\mathscr{K}[\mathbf{r}_1], \mathscr{K}[\mathbf{r}_2], ...$, SLE (55) must be formed and solved for each $\mathbf{r}$. In contrast with globally-supported bases which have large, dense collocation matrices, local support gives rise to small SLEs. The intent is to reduce computing and storage costs at the expense of a small decrease in accuracy.

It is interesting to note the similarity between SLEs (22) and (55). This similarity suggests that the RBF scattered data interpolation problem class (with an augmenting polynomial basis) may be interpreted as a QP problem. In particular, the RBF weight factors $\mathbf{w}$ represent the primal variables whereas the polynomial coefficients $\boldsymbol{\lambda}$ represent the dual variables/Lagrange multipliers. Perhaps most profoundly, the equivalence of (22) and (55) suggests that the duality concept underpins scattered data interpolation, as it does so many other problem classes in computational engineering.

3.1.2 Hermite Interpolation

Hermite RBF interpolation builds upon the standard RBF approach by expanding $s(\mathbf{r})$ as linear combinations of both the RBKs and their spatial derivatives [18][25][26][27][28]. To the author's knowledge there is no limit to the order of derivatives which may be included in the basis, however only first-order derivatives will be explored here. Well-posedness of the Hermite interpolation problem requires that the solution data $\mathscr{F}$ be complemented by derivative data $\mathscr{F}'$. In this way, Hermite interpolation offers a potential improvement in accuracy of both the interpolant and its derivatives by encoding more of the available data.

In the theory that follows, an m=2-dimensional embedding space is considered for simplicity; generalization to higher-dimensional spaces is intuitive. The sets $\mathscr{A}$ and $\mathscr{F}$, which contain the given data centers and data field respectively, are supplemented by the following:

- A set of data centers $\mathscr{B} = [\boldsymbol{\beta}_1 \ \cdots \ \boldsymbol{\beta}_i \ \cdots \ \boldsymbol{\beta}_{M_x}]^{\mathrm{T}}$, wherein the x-derivative data $\mathscr{F}_x = \left[\dfrac{\partial f_1}{\partial x} \ \cdots \ \dfrac{\partial f_i}{\partial x} \ \cdots \ \dfrac{\partial f_{M_x}}{\partial x}\right]^{\mathrm{T}}$ are given.

- A set of data centers $\mathscr{G} = [\boldsymbol{\gamma}_1 \ \cdots \ \boldsymbol{\gamma}_i \ \cdots \ \boldsymbol{\gamma}_{M_y}]^{\mathrm{T}}$, wherein the y-derivative data $\mathscr{F}_y = \left[\dfrac{\partial f_1}{\partial y} \ \cdots \ \dfrac{\partial f_i}{\partial y} \ \cdots \ \dfrac{\partial f_{M_y}}{\partial y}\right]^{\mathrm{T}}$ are given.

With the incorporation of these two sets of data centers, the k-stencil introduced earlier must be redefined. In particular, the subsets $\mathscr{K}_x[\mathbf{r}] \subseteq \mathscr{B}$ and $\mathscr{K}_y[\mathbf{r}] \subseteq \mathscr{G}$ include the respective k_x x-derivative centers and k_y y-derivative centers nearest to $\mathbf{r}$. Two subsets of the derivative data are defined:

- $\mathscr{F}_x[\mathbf{r}] \subset \mathscr{F}_x$: the subset of x-derivative data located at the data centers in $\mathscr{K}_x[\mathbf{r}]$.
- $\mathscr{F}_y[\mathbf{r}] \subset \mathscr{F}_y$: the subset of y-derivative data located at the data centers in $\mathscr{K}_y[\mathbf{r}]$.

The Hermite expansion of $s(\mathbf{r})$ incorporates the additional data sets by way of an extended basis (cf. Equation (52))

$$s(\mathbf{r}) = \sum_{\boldsymbol{\alpha}\in\mathscr{K}[\mathbf{r}]} w(\boldsymbol{\alpha})\phi(\mathbf{r},\boldsymbol{\alpha}) + \sum_{\boldsymbol{\beta}\in\mathscr{K}_x[\mathbf{r}]} w_x(\boldsymbol{\beta})\frac{\partial\phi(\mathbf{r},\boldsymbol{\beta})}{\partial x} + \sum_{\boldsymbol{\gamma}\in\mathscr{K}_y[\mathbf{r}]} w_y(\boldsymbol{\gamma})\frac{\partial\phi(\mathbf{r},\boldsymbol{\gamma})}{\partial y} + \sum_{i=1}^{z}\lambda_i p_i(\mathbf{r}) \tag{56}$$

where $w_{\mathrm{x}}(\boldsymbol{\beta}), w_{\mathrm{y}}(\boldsymbol{\gamma}) \in \mathbb{R}$ are the scalar weight factors for the x and y-derivatives of the RBKs. These weights follow from collocation of the given derivative data and the x and y-derivatives of the interpolant at their respective data centers. These collocation constraint sets may be summarized as follows

$$\left\{\begin{array}{l} \left.\dfrac{\partial s(\mathbf{r})}{\partial x}\right|_{\mathbf{r}=\boldsymbol{\beta}} = \dfrac{\partial f}{\partial x},\ \forall\boldsymbol{\beta}\in\mathscr{K}_x[\mathbf{r}],\ \forall\dfrac{\partial f}{\partial x}\in\mathscr{F}_x[\mathbf{r}] \\ \left.\dfrac{\partial s(\mathbf{r})}{\partial y}\right|_{\mathbf{r}=\boldsymbol{\gamma}} = \dfrac{\partial f}{\partial y},\ \forall\boldsymbol{\gamma}\in\mathscr{K}_y[\mathbf{r}],\ \forall\dfrac{\partial f}{\partial y}\in\mathscr{F}_y[\mathbf{r}] \end{array}\right\} \tag{57}$$

Combination of Equations (55) and (57) gives a SLE of size $(k + k_x + k_y + z)$ x $(k + k_x + k_y + z)$

$$\begin{bmatrix} \boldsymbol{\Phi}(\mathscr{K}[\mathbf{r}],\mathscr{K}[\mathbf{r}]) & \boldsymbol{\Phi}_x(\mathscr{K}[\mathbf{r}],\mathscr{K}_x[\mathbf{r}]) & \boldsymbol{\Phi}_y(\mathscr{K}[\mathbf{r}],\mathscr{K}_y[\mathbf{r}]) & \mathbf{P}(\mathscr{K}[\mathbf{r}]) \\ \boldsymbol{\Phi}_x(\mathscr{K}_x[\mathbf{r}],\mathscr{K}[\mathbf{r}]) & \boldsymbol{\Phi}_{xx}(\mathscr{K}_x[\mathbf{r}],\mathscr{K}_x[\mathbf{r}]) & \boldsymbol{\Phi}_{xy}(\mathscr{K}_x[\mathbf{r}],\mathscr{K}_y[\mathbf{r}]) & \mathbf{P}_x(\mathscr{K}_x[\mathbf{r}]) \\ \boldsymbol{\Phi}_y(\mathscr{K}_y[\mathbf{r}],\mathscr{K}[\mathbf{r}]) & \boldsymbol{\Phi}_{yx}(\mathscr{K}_y[\mathbf{r}],\mathscr{K}_x[\mathbf{r}]) & \boldsymbol{\Phi}_{yy}(\mathscr{K}_y[\mathbf{r}],\mathscr{K}_y[\mathbf{r}]) & \mathbf{P}_y(\mathscr{K}_y[\mathbf{r}]) \\ \mathbf{P}(\mathscr{K}[\mathbf{r}])^{\mathrm{T}} & \mathbf{P}_x(\mathscr{K}_x[\mathbf{r}])^{\mathrm{T}} & \mathbf{P}_y(\mathscr{K}_y[\mathbf{r}])^{\mathrm{T}} & \mathbf{0}^{zxz} \end{bmatrix} \begin{bmatrix} \mathbf{w} \\ \mathbf{w}_x \\ \mathbf{w}_y \\ \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} \mathscr{F}[\mathbf{r}] & \mathscr{F}_x[\mathbf{r}] & \mathscr{F}_y[\mathbf{r}] & \mathbf{0}^z \end{bmatrix}^{\mathrm{T}} \tag{58}$$

where the sub-matrices above contain the partial derivatives of the RBKs and polynomial as indicated by the subscripts. Appendix B may be consulted for details on how these sub-matrices are generated.

The cumbersome looking collocation matrix in Equation (58) may be partially simplified by sampling the data in the same quantities and locations, such that $M = M_x = M_y$ and $\mathscr{A} = \mathscr{B} = \mathscr{G}$. As a consequence, the stencils at $\mathbf{r}$ will be identical with $\mathscr{K}[\mathbf{r}] = \mathscr{K}_x[\mathbf{r}] = \mathscr{K}_y[\mathbf{r}]$. Further simplifications result from the following mathematical properties of RBKs:

- Skew-symmetry of $\boldsymbol{\Phi}_x$ and $\boldsymbol{\Phi}_y$, which implies $\boldsymbol{\Phi}_x(\mathscr{K}[\mathbf{r}],\mathscr{K}_x[\mathbf{r}]) = -\boldsymbol{\Phi}_x(\mathscr{K}_x[\mathbf{r}],\mathscr{K}[\mathbf{r}])$ and $\boldsymbol{\Phi}_y(\mathscr{K}[\mathbf{r}],\mathscr{K}_y[\mathbf{r}]) = -\boldsymbol{\Phi}_y(\mathscr{K}_y[\mathbf{r}],\mathscr{K}[\mathbf{r}])$.
- Symmetry of $\boldsymbol{\Phi}, \boldsymbol{\Phi}_{xx}, \boldsymbol{\Phi}_{yy},$ and $\boldsymbol{\Phi}_{xy}$, which implies $\boldsymbol{\Phi}_{xy}(\mathscr{K}_x[\mathbf{r}],\mathscr{K}_y[\mathbf{r}]) = \boldsymbol{\Phi}_{yx}(\mathscr{K}_y[\mathbf{r}],\mathscr{K}_x[\mathbf{r}])$.

In unison, the circumstances described above allow Equation (58) to be recast as a symmetric SLE

$$\begin{bmatrix} \boldsymbol{\Phi} & \boldsymbol{\Phi}_x & \boldsymbol{\Phi}_y & \mathbf{P} \\ -\boldsymbol{\Phi}_x & \boldsymbol{\Phi}_{xx} & \boldsymbol{\Phi}_{xy} & \mathbf{P}_x \\ -\boldsymbol{\Phi}_y & \boldsymbol{\Phi}_{xy} & \boldsymbol{\Phi}_{yy} & \mathbf{P}_y \\ \mathbf{P}^{\mathrm{T}} & \mathbf{P}_x^{\mathrm{T}} & \mathbf{P}_y^{\mathrm{T}} & \mathbf{0}^{z\mathrm{x}z} \end{bmatrix} \begin{bmatrix} \mathbf{w} \\ \mathbf{w}_x \\ \mathbf{w}_y \\ \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} \mathscr{F} \\ \mathscr{F}_x \\ \mathscr{F}_y \\ \mathbf{0}^z \end{bmatrix} \tag{59}$$

where the $(\bullet, \bullet)$ kernel notation used previously has been dropped as a reminder that all sub-matrices in Equation (59) are centered and evaluated at the coincident data and derivative centers. Once again assuming the collocation matrix to be invertible, substitution of $\mathbf{w},\ \mathbf{w}_x,\ \mathbf{w}_y$, and $\boldsymbol{\lambda}$ from SLE (59) into Equation (56) gives $s(\mathbf{r})$.

3.2 Discretization of PDEs

A relatively novel application of the RBF framework is the discretization of ordinary and partial differential equations. Consider an open domain $\Omega \subset \mathbb{R}^3$ with a closed boundary $\Gamma = \partial\Omega$. As in Problem (12), Γ is composed of non-overlapping Dirichlet and Neumann sub-boundaries $\Gamma_{\mathfrak{D}}$ and $\Gamma_{\mathfrak{N}}$, respectively. Associated with this domain is a (assumed) unique, continuous scalar field[7] $u(\mathbf{r})$ which satisfies Equation (60)

$$\begin{aligned} \mathscr{L}u(\mathbf{r}) &= f(\mathbf{r}), \quad \forall \mathbf{r} \in \Omega \\ u(\mathbf{r}) &= g(\mathbf{r}), \quad \forall \mathbf{r} \in \Gamma_{\mathfrak{D}} \\ \nabla u(\mathbf{r}) &= h(\mathbf{r}), \quad \forall \mathbf{r} \in \Gamma_{\mathfrak{N}} \end{aligned} \tag{60}$$

where $f(\mathbf{r}) : \Omega \mapsto \mathbb{R}, g(\mathbf{r}) : \Gamma_{\mathfrak{D}} \mapsto \mathbb{R}, h(\mathbf{r}) : \Gamma_{\mathfrak{N}} \mapsto \mathbb{R}$ are prescribed scalar fields and $\mathscr{L}$ represents a continuous differential operator (e.g. gradient, Laplacian, advection-diffusion, Helmholtz, etc.).

[7] The concepts presented in this section generalize straightforwardly to vector and tensor fields.

In fitting with the rest of this chapter, Ω and Γ are discretized with a set of scattered data centers $\mathscr{A} = [\boldsymbol{\alpha}_1 \ \cdots \ \boldsymbol{\alpha}_i \ \cdots \ \boldsymbol{\alpha}_M]^{\mathrm{T}}$. By placing an RBF at each of these data centers, u may be approximated at an arbitrary $\mathbf{r} \in \Omega$, with locally-supported stencil $\mathscr{K}[\mathbf{r}]$, by a linear combination of RBKs and a d^{th} degree polynomial

$$u(\mathbf{r}) \approx \sum_{\boldsymbol{\alpha} \in \mathscr{K}[\mathbf{r}]} w(\boldsymbol{\alpha}) \phi(\mathbf{r}, \boldsymbol{\alpha}) + \sum_{i=1}^{z} \lambda_i p_i(\mathbf{r}) \tag{61}$$

Collocation requires that Equation (61) be satisfied at each data center contained in $\mathscr{K}[\mathbf{r}]$, such that $\left\{ u(\mathbf{r})\Big|_{\mathbf{r}=\boldsymbol{\alpha}} = u(\boldsymbol{\alpha}), \ \forall \boldsymbol{\alpha} \in \mathscr{K}[\mathbf{r}] \right\}$. These $u(\boldsymbol{\alpha})$, which are treated as unknown, are entered into a column vector $\mathbf{u}(\mathscr{K}[\mathbf{r}])$ of length k. In conjunction with the vanishing moment constraint set defined by Equation (54), collocation yields the following SLE of size $(k+z)$ x $(k+z)$

$$\begin{bmatrix} \boldsymbol{\Phi}(\mathscr{K}[\mathbf{r}], \mathscr{K}[\mathbf{r}]) & \mathbf{P}(\mathscr{K}[\mathbf{r}]) \\ \mathbf{P}(\mathscr{K}[\mathbf{r}])^{\mathrm{T}} & \mathbf{0}^{z\mathrm{x}z} \end{bmatrix} \begin{bmatrix} \mathbf{w} \\ \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} \mathbf{u}(\mathscr{K}[\mathbf{r}]) \\ \mathbf{0}^{z} \end{bmatrix} \tag{62}$$

SLE (62) implies that the BCs on $\Gamma_{\mathfrak{D}}$ and $\Gamma_{\mathfrak{N}}$ will not necessarily be satisfied. Subsection 3.2.1 will explain how this violation is resolved if one or more of the data centers contained in $\mathscr{K}[\mathbf{r}]$ lie on the boundary.

The RBK and polynomial weighting factors may be expressed in terms of the unknown $\mathbf{u}(\mathscr{K}[\mathbf{r}])$ by solution of (62). For now, the collocation matrix $\mathbf{B}(\mathscr{K}[\mathbf{r}])$ is naively assumed to be invertible

$$\begin{bmatrix} \mathbf{w} \\ \boldsymbol{\lambda} \end{bmatrix} = \mathbf{B}(\mathscr{K}[\mathbf{r}])^{-1} \begin{bmatrix} \mathbf{u}(\mathscr{K}[\mathbf{r}]) \\ \mathbf{0}^{z} \end{bmatrix} \tag{63}$$

Returning now to Equation (61), application of $\mathscr{L}$ to $u(\mathbf{r})$ produces the following

$$\mathscr{L}u(\mathbf{r}) = \sum_{\boldsymbol{\alpha}\in\mathscr{K}[\mathbf{r}]} w(\boldsymbol{\alpha})(\mathscr{L}\phi(\mathbf{r},\boldsymbol{\alpha})) + \sum_{i=1}^{z} \lambda_i(\mathscr{L}p_i(\mathbf{r})) \tag{64}$$

Recasting the equation above in block matrix form, followed by substitution of Equation (63) for $\mathbf{w}$ and $\boldsymbol{\lambda}$, gives

$$\begin{aligned}\mathscr{L}u(\mathbf{r}) &= \underbrace{\begin{bmatrix}\mathscr{L}\boldsymbol{\phi}(\mathbf{r},\mathscr{K}[\mathbf{r}]) & \mathscr{L}\mathbf{p}(\mathbf{r})\end{bmatrix}\mathbf{B}(\mathscr{K}[\mathbf{r}])^{-1}}_{=\begin{bmatrix}\boldsymbol{\ell}^{\phi}(\mathbf{r},\mathscr{K}[\mathbf{r}]) & \boldsymbol{\ell}^{p}(\mathbf{r})\end{bmatrix}} \begin{bmatrix}\mathbf{u}(\mathscr{K}[\mathbf{r}]) \\ \\ \mathbf{0}^{z}\end{bmatrix}\end{aligned} \tag{65}$$

where $\mathscr{L}\boldsymbol{\phi}(\mathbf{r},\mathscr{K}[\mathbf{r}])$ is a length k row vector with entries $\mathscr{L}\phi(\mathbf{r},\boldsymbol{\alpha}),\ \forall\boldsymbol{\alpha}\in\mathscr{K}[\mathbf{r}]$ and $\mathscr{L}\mathbf{p}(\mathbf{r})$ is a length z row vector with entries $\mathscr{L}p_{i=1,\ldots,z}(\mathbf{r})$. Making use of the $\begin{bmatrix}\boldsymbol{\ell}^{\phi}(\mathbf{r},\mathscr{K}[\mathbf{r}]) & \boldsymbol{\ell}^{p}(\mathbf{r})\end{bmatrix}$ vector defined above, (65) reduces to the following summation

$$\begin{aligned}\mathscr{L}u(\mathbf{r}) &= \boldsymbol{\ell}^{\phi}(\mathbf{r},\mathscr{K}[\mathbf{r}])\mathbf{u}(\mathscr{K}[\mathbf{r}]) + \boldsymbol{\ell}^{p}(\mathbf{r})\mathbf{0}^{z} \\ &= \sum_{\boldsymbol{\alpha}\in\mathscr{K}[\mathbf{r}]} \ell^{\phi}(\mathbf{r},\boldsymbol{\alpha})u(\boldsymbol{\alpha})\end{aligned} \tag{66}$$

The length k row vector $\boldsymbol{\ell}^{\phi}(\mathbf{r},\mathscr{K}[\mathbf{r}])$ resembles a scalar field of weight factors on the data centers nearest to $\mathbf{r}$; its entries consist of $\ell^{\phi}(\mathbf{r},\boldsymbol{\alpha}),\ \forall\boldsymbol{\alpha}\in\mathscr{K}[\mathbf{r}]$. Thus, Equation (66) is a *local representation* of $\mathscr{L}u(\mathbf{r})$ as a linear combination of the $u(\boldsymbol{\alpha})$ nearest to $\mathbf{r}$. An equivalent *global representation* is achieved by appending the $u(\boldsymbol{\alpha})$ outside of $\mathscr{K}[\mathbf{r}]$ to the summation above, each with a weight of zero. For this purpose, the indicator function $\{\boldsymbol{\alpha},\mathscr{K}[\mathbf{r}]\}$ is introduced

$$\{\boldsymbol{\alpha},\mathscr{K}[\mathbf{r}]\} = \begin{cases}1,\ \boldsymbol{\alpha}\in\mathscr{K}[\mathbf{r}] \\ \\ 0,\ \boldsymbol{\alpha}\notin\mathscr{K}[\mathbf{r}]\end{cases}$$

The indicator function enables the following global representation of $\mathscr{L}u(\mathbf{r})$

$$\mathscr{L}u(\mathbf{r}) = \sum_{\boldsymbol{\alpha}\in\mathscr{A}} \{\boldsymbol{\alpha}, \mathscr{K}[\mathbf{r}]\}\ell^{\phi}(\mathbf{r},\boldsymbol{\alpha})u(\boldsymbol{\alpha}) \tag{67}$$

Observe that the summation above has been extended over the entire set of data centers $\mathscr{A}$ as opposed to just the set of centers $\mathscr{K}[\mathbf{r}]$ nearest to $\mathbf{r}$. With $\mathscr{L}u(\mathbf{r})$ now in global form, the following steps are taken:

1. $\mathscr{K}[\boldsymbol{\alpha}_1]$ is assembled from the k data centers nearest to *and including* $\boldsymbol{\alpha}_1$.
2. The collocation matrix $\mathbf{B}(\mathscr{K}[\boldsymbol{\alpha}_1])$ and weight vector $\boldsymbol{\ell}^{\phi}(\mathbf{r},\mathscr{K}[\boldsymbol{\alpha}_1])\Big|_{\mathbf{r}=\boldsymbol{\alpha}_1}$ are assembled via Equations (62) and (65), respectively.
3. Substitution of $\boldsymbol{\ell}^{\phi}(\mathbf{r},\mathscr{K}[\boldsymbol{\alpha}_1])\Big|_{\mathbf{r}=\boldsymbol{\alpha}_1}$ into Equation (67) gives $\mathscr{L}u(\mathbf{r})\Big|_{\mathbf{r}=\boldsymbol{\alpha}_1}$ as a weighted sum of the unknown $u(\boldsymbol{\alpha}_1),...,u(\boldsymbol{\alpha}_M)$. $\mathscr{L}u(\mathbf{r})\Big|_{\mathbf{r}=\boldsymbol{\alpha}_1}$ may then be equated with the given datum $f(\mathbf{r})\Big|_{\mathbf{r}=\boldsymbol{\alpha}_1}$ in accordance with Equation (60).
4. Steps one through three are repeated for each data center $\boldsymbol{\alpha}_2,...,\boldsymbol{\alpha}_M$.

This sequence of steps leads to the SLE of size M x M appearing below

$$\begin{aligned}
\sum_{\boldsymbol{\alpha}\in\mathscr{A}} \{\boldsymbol{\alpha}, \mathscr{K}[\boldsymbol{\alpha}_1]\}\ell^{\phi}(\mathbf{r},\boldsymbol{\alpha})\Big|_{\mathbf{r}=\boldsymbol{\alpha}_1} u(\boldsymbol{\alpha}) &= f(\mathbf{r})\Big|_{\mathbf{r}=\boldsymbol{\alpha}_1} \\
\vdots & \quad \vdots \\
\sum_{\boldsymbol{\alpha}\in\mathscr{A}} \{\boldsymbol{\alpha}, \mathscr{K}[\boldsymbol{\alpha}_i]\}\ell^{\phi}(\mathbf{r},\boldsymbol{\alpha})\Big|_{\mathbf{r}=\boldsymbol{\alpha}_i} u(\boldsymbol{\alpha}) &= f(\mathbf{r})\Big|_{\mathbf{r}=\boldsymbol{\alpha}_i} \\
\vdots & \quad \vdots \\
\sum_{\boldsymbol{\alpha}\in\mathscr{A}} \{\boldsymbol{\alpha}, \mathscr{K}[\boldsymbol{\alpha}_M]\}\ell^{\phi}(\mathbf{r},\boldsymbol{\alpha})\Big|_{\mathbf{r}=\boldsymbol{\alpha}_M} u(\boldsymbol{\alpha}) &= f(\mathbf{r})\Big|_{\mathbf{r}=\boldsymbol{\alpha}_M}
\end{aligned} \tag{68}$$

Where the right-hand side $f(\mathbf{r})\Big|_{\mathbf{r}=\boldsymbol{\alpha}_1}, ..., f(\mathbf{r})\Big|_{\mathbf{r}=\boldsymbol{\alpha}_M}$ represents the prescribed scalar field on the set of data centers $\mathscr{A}$. Alternatively, in the more familiar vector-matrix $\mathbf{Ax} = \mathbf{b}$ format, SLE (68) may be expressed as

$$\underbrace{\begin{bmatrix} \{\boldsymbol{\alpha}_1, \mathscr{K}[\boldsymbol{\alpha}_1]\}\ell^{\phi}(\mathbf{r}, \boldsymbol{\alpha}_1)\Big|_{\mathbf{r}=\boldsymbol{\alpha}_1} & \cdots & \{\boldsymbol{\alpha}_M, \mathscr{K}[\boldsymbol{\alpha}_1]\}\ell^{\phi}(\mathbf{r}, \boldsymbol{\alpha}_M)\Big|_{\mathbf{r}=\boldsymbol{\alpha}_1} \\ \vdots & \ddots & \vdots \\ \{\boldsymbol{\alpha}_1, \mathscr{K}[\boldsymbol{\alpha}_M]\}\ell^{\phi}(\mathbf{r}, \boldsymbol{\alpha}_1)\Big|_{\mathbf{r}=\boldsymbol{\alpha}_M} & \cdots & \{\boldsymbol{\alpha}_M, \mathscr{K}[\boldsymbol{\alpha}_M]\}\ell^{\phi}(\mathbf{r}, \boldsymbol{\alpha}_M)\Big|_{\mathbf{r}=\boldsymbol{\alpha}_M} \end{bmatrix}}_{=\mathbf{L}} \underbrace{\begin{bmatrix} u(\boldsymbol{\alpha}_1) \\ \vdots \\ u(\boldsymbol{\alpha}_M) \end{bmatrix}}_{=\mathbf{u}(\mathscr{A})}$$

$$= \underbrace{\begin{bmatrix} f(\mathbf{r})\Big|_{\mathbf{r}=\boldsymbol{\alpha}_1} & \cdots & f(\mathbf{r})\Big|_{\mathbf{r}=\boldsymbol{\alpha}_M} \end{bmatrix}^{\mathrm{T}}}_{=\mathbf{f}(\mathscr{A})} \tag{69}$$

At this stage, the RBF PDE discretization is, in a fundamental sense, identical to the conventional FDM/FEM/FVM and concludes with the solution of the global SLE, $\mathbf{Lu}(\mathscr{A}) = \mathbf{f}(\mathscr{A})$. $\mathbf{L}$ represents the discrete form of the continuous $\mathscr{L}$ operator and, for a one-dimensional domain $\Omega \subset \mathbb{R}$, takes the form of a band matrix with bandwidth k as shown below

$$\mathbf{L} = \begin{bmatrix} \boldsymbol{\ell}^{\phi}(\mathbf{r}, \mathscr{K}[\boldsymbol{\alpha}_1])\Big|_{\mathbf{r}=\boldsymbol{\alpha}_1} & & & & \\ & \ddots & & & \\ & & \boldsymbol{\ell}^{\phi}(\mathbf{r}, \mathscr{K}[\boldsymbol{\alpha}_i])\Big|_{\mathbf{r}=\boldsymbol{\alpha}_i} & & \\ & & & \ddots & \\ & & & & \boldsymbol{\ell}^{\phi}(\mathbf{r}, \mathscr{K}[\boldsymbol{\alpha}_M])\Big|_{\mathbf{r}=\boldsymbol{\alpha}_M} \end{bmatrix} \tag{70}$$

While this band matrix structure occurs naturally in one dimension, care must be taken with the indexing of the data centers in higher dimensions if such a structure is to be preserved [17]. For small stencils with $k \ll M$, $\mathbf{L}$ becomes sparse and the solution of SLE (69) can be done efficiently.

To conclude the discussion of RBF discretization of PDEs, we revisit Equation (67) which has an implicit, yet valuable capability. First, each of the $u(\boldsymbol{\alpha})$ in the summation are redefined as known input data (i.e. a given scalar field on the data center set $\mathscr{A}$). With this modification, Equation (67) *interpolates* $\mathscr{L}u(\mathbf{r})$ as a global, linear combination of $u(\boldsymbol{\alpha}_1), ..., u(\boldsymbol{\alpha}_M)$; in effect, RBF discretization is coupled with the scattered data interpolation methodology of Subsections 3.1.1 and 3.1.2. This capability will be used extensively in Chapters 5 and 6, in particular to estimate the gradients of temperature and stress fields.

3.2.1 Boundary Conditions

To enforce the BCs in Equation (60), the relevant entries of $\mathbf{L}$ must be appropriately modified. For the Dirichlet BC, a continuous field $g(\mathbf{r})$ is prescribed on $\Gamma_{\mathfrak{D}}$, such that $u(\mathbf{r}) = g(\mathbf{r}),\ \forall \mathbf{r} \in \Gamma_{\mathfrak{D}}$. Given an arbitrary data center $\boldsymbol{\alpha}_j \in \Gamma_{\mathfrak{D}}$ with locally-supported stencil $\mathscr{K}[\boldsymbol{\alpha}_j]$, the Dirichlet BC requires

$$u(\mathbf{r})\Big|_{\mathbf{r}=\boldsymbol{\alpha}_j} = \sum_{\boldsymbol{\alpha}\in\mathscr{A}} \{\boldsymbol{\alpha}, \mathscr{K}[\boldsymbol{\alpha}_j]\}\ell^{\phi}(\mathbf{r}, \boldsymbol{\alpha})\Big|_{\mathbf{r}=\boldsymbol{\alpha}_j} u(\boldsymbol{\alpha}) = g(\mathbf{r})\Big|_{\mathbf{r}=\boldsymbol{\alpha}_j} \tag{71}$$

Simplification of (71) may be achieved by expanding the summation and moving $g(\mathbf{r})\Big|_{\mathbf{r}=\boldsymbol{\alpha}_j}$ to the left-hand side

$$\begin{aligned}
&\{\boldsymbol{\alpha}_1, \mathscr{K}[\boldsymbol{\alpha}_j]\}\ell^{\phi}(\mathbf{r}, \boldsymbol{\alpha}_1)\Big|_{\mathbf{r}=\boldsymbol{\alpha}_j} u(\boldsymbol{\alpha}_1) \ +\ \{\boldsymbol{\alpha}_2, \mathscr{K}[\boldsymbol{\alpha}_j]\}\ell^{\phi}(\mathbf{r}, \boldsymbol{\alpha}_2)\Big|_{\mathbf{r}=\boldsymbol{\alpha}_j} u(\boldsymbol{\alpha}_2) + \cdots + \\
&\Big(\{\boldsymbol{\alpha}_j, \mathscr{K}[\boldsymbol{\alpha}_j]\}\ell^{\phi}(\mathbf{r}, \boldsymbol{\alpha}_j)\Big|_{\mathbf{r}=\boldsymbol{\alpha}_j} u(\boldsymbol{\alpha}_j) - g(\mathbf{r})\Big|_{\mathbf{r}=\boldsymbol{\alpha}_j}\Big) + \cdots + \\
&\{\boldsymbol{\alpha}_{M-1}, \mathscr{K}[\boldsymbol{\alpha}_j]\}\ell^{\phi}(\mathbf{r}, \boldsymbol{\alpha}_{M-1})\Big|_{\mathbf{r}=\boldsymbol{\alpha}_j} u(\boldsymbol{\alpha}_{M-1}) \ +\ \{\boldsymbol{\alpha}_M, \mathscr{K}[\boldsymbol{\alpha}_j]\}\ell^{\phi}(\mathbf{r}, \boldsymbol{\alpha}_M)\Big|_{\mathbf{r}=\boldsymbol{\alpha}_j} u(\boldsymbol{\alpha}_M) = 0
\end{aligned} \tag{72}$$

Recalling that $\{\boldsymbol{\alpha}_j, \mathscr{K}[\boldsymbol{\alpha}_j]\} = 1$ (by definition of a k-stencil) and $u(\boldsymbol{\alpha}_j) = g(\mathbf{r})\Big|_{\mathbf{r}=\boldsymbol{\alpha}_j}$, Equation (72) is satisfied by $\ell^{\phi}(\mathbf{r}, \boldsymbol{\alpha}_i)\Big|_{\mathbf{r}=\boldsymbol{\alpha}_j} = \delta_{i,j}$ for $i = 1, ..., M$. In other words, the

Dirichlet BCs are enforced by making the following modifications to SLE (69):

1. For a data center $\boldsymbol{\alpha}_j$ lying on $\Gamma_{\mathfrak{D}}$, row j of $\mathbf{L}$ is replaced with the entries $\mathbf{L}_{j,i=j} = 1$ and $\mathbf{L}_{j,i\neq j} = 0$ for $i = 1, ..., M$. $M - k$ of the entries in row j of $\mathbf{L}$ will already be zero as they correspond to data centers that lie outside of $\mathscr{K}[\boldsymbol{\alpha}_j]$.
2. Row j of $\mathbf{f}(\mathscr{A})$ is replaced with $g(\mathbf{r})\Big|_{\mathbf{r}=\boldsymbol{\alpha}_j}$.
3. Steps one and two are repeated for each of the data centers lying on $\Gamma_{\mathfrak{D}}$.

Analysis of the Neumann sub-boundary is not as straightforward since it involves a constraint on $\nabla u(\mathbf{r})$ (natural BC) as opposed to $u(\mathbf{r})$ (essential BC). For a total of $M_{\mathfrak{N}}$ data centers lying on $\Gamma_{\mathfrak{N}}$, this means that an equal number of equations must be added for closure of SLE (69). These supplementary equations are enabled by a set of $M_{\mathfrak{N}}$ *ghost centers*, $\mathscr{A}^G = [\boldsymbol{\alpha}_1^G \ \cdots \ \boldsymbol{\alpha}_i^G \ \cdots \ \boldsymbol{\alpha}_{M_{\mathfrak{N}}}^G]^{\mathrm{T}}$, which are placed in the vicinity of $\Gamma_{\mathfrak{N}}$ and outside of Ω. With the introduction of the ghost centers, the modified set of data centers becomes $\mathscr{A}^{\dagger} = \mathscr{A} \cup \mathscr{A}^G$. Moreover, an additional RBF is centered at each $\boldsymbol{\alpha}^G \in \mathscr{A}^{\dagger}$ which in turn requires modification of the SLE (69).

As was done with the Dirichlet BC, consider a continuous field $h(\mathbf{r})$ prescribed on $\Gamma_{\mathfrak{N}}$, such that $\nabla u(\mathbf{r}) = h(\mathbf{r}), \ \forall \mathbf{r} \in \Gamma_{\mathfrak{N}}$. Given an arbitrary data center $\boldsymbol{\alpha}_j \in \Gamma_{\mathfrak{N}}$ with locally-supported stencil $\mathscr{K}[\boldsymbol{\alpha}_j]$, an approximation of $\nabla u(\mathbf{r})\Big|_{\mathbf{r}=\boldsymbol{\alpha}_j}$ is required. A local approximation can be obtained by the substitution of $\mathscr{L} \Rightarrow \nabla$ in Equation (65), which gives

$$\nabla u(\mathbf{r})\Big|_{\mathbf{r}=\boldsymbol{\alpha}_j} = \underbrace{\left[\nabla \boldsymbol{\phi}(\mathbf{r}, \mathscr{K}[\boldsymbol{\alpha}_j])\Big|_{\mathbf{r}=\boldsymbol{\alpha}_j} \quad \nabla \mathbf{p}(\mathbf{r})\Big|_{\mathbf{r}=\boldsymbol{\alpha}_j}\right] \mathbf{B}(\mathscr{K}[\boldsymbol{\alpha}_j])^{-1}}_{= \left[\mathbf{d}^{\phi}(\mathbf{r}, \mathscr{K}[\boldsymbol{\alpha}_j])\Big|_{\mathbf{r}=\boldsymbol{\alpha}_j} \quad \mathbf{d}^{p}(\mathbf{r})\Big|_{\mathbf{r}=\boldsymbol{\alpha}_j}\right]} \begin{bmatrix} \mathbf{u}(\mathscr{K}[\boldsymbol{\alpha}_j]) \\ \\ \mathbf{0}^z \end{bmatrix} \tag{73}$$

where $\nabla\boldsymbol{\phi}(\mathbf{r},\mathscr{K}[\boldsymbol{\alpha}_j])\Big|_{\mathbf{r}=\boldsymbol{\alpha}_j}$ is a length k row vector with entries $\nabla\phi(\mathbf{r},\boldsymbol{\alpha})\Big|_{\mathbf{r}=\boldsymbol{\alpha}_j}$, $\forall\boldsymbol{\alpha}\in\mathscr{K}[\boldsymbol{\alpha}_j]$ and $\nabla\mathbf{p}(\mathbf{r})\Big|_{\mathbf{r}=\boldsymbol{\alpha}_j}$ is a length z row vector with entries $\nabla p_{i=1,\ldots,z}(\mathbf{r})\Big|_{\mathbf{r}=\boldsymbol{\alpha}_j}$. In the equivalent global representation of Equation (73), the Neumann BC requires

$$\nabla u(\mathbf{r})\Big|_{\mathbf{r}=\boldsymbol{\alpha}_j}=\sum_{\boldsymbol{\alpha}\in\mathscr{A}^\dagger}\{\boldsymbol{\alpha},\mathscr{K}[\boldsymbol{\alpha}_j]\}d^\phi(\mathbf{r},\boldsymbol{\alpha})\Big|_{\mathbf{r}=\boldsymbol{\alpha}_j}u(\boldsymbol{\alpha})=h(\mathbf{r})\Big|_{\mathbf{r}=\boldsymbol{\alpha}_j}\tag{74}$$

such that the gradient is approximated by a linear combination of the $u(\boldsymbol{\alpha})$ with scalar weights $\{\boldsymbol{\alpha},\mathscr{K}[\boldsymbol{\alpha}_j]\}d^\phi(\mathbf{r},\boldsymbol{\alpha})\Big|_{\mathbf{r}=\boldsymbol{\alpha}_j}$. With respect to SLE (69), the following measures are taken to enforce condition (74):

1. For a data center $\boldsymbol{\alpha}_j$ lying on $\Gamma_\mathfrak{N}$, a ghost center and accompanying RBF are created near $\boldsymbol{\alpha}_j$ and outside of Ω.

2. $\mathscr{K}[\boldsymbol{\alpha}_j]$ is assembled from the k data centers and ghost centers nearest to and including $\boldsymbol{\alpha}_j$.

3. The collocation matrix $\mathbf{B}(\mathscr{K}[\boldsymbol{\alpha}_j])$ and weight vector $\mathbf{d}^\phi(\mathbf{r},\mathscr{K}[\boldsymbol{\alpha}_j])\Big|_{\mathbf{r}=\boldsymbol{\alpha}_j}$ are assembled via Equations (62) and (73), respectively.

4. As per (74), row j of $\mathbf{L}$ is replaced with entries $\mathbf{L}_{j,i}=\{\boldsymbol{\alpha}_i,\mathscr{K}[\boldsymbol{\alpha}_j]\}d^\phi(\mathbf{r},\boldsymbol{\alpha}_i)\Big|_{\mathbf{r}=\boldsymbol{\alpha}_j}$ for $i=1,\ldots,M+M_\mathfrak{N}$.

5. Row j of $\mathbf{f}(\mathscr{A}^\dagger)$ is replaced with $h(\mathbf{r})\Big|_{\mathbf{r}=\boldsymbol{\alpha}_j}$.

6. Steps one through five are repeated for each of the $M_\mathfrak{N}$ data centers lying on $\Gamma_\mathfrak{N}$.

In practice, step one is done in pre-processing wherein $\mathbf{L}$ is pre-allocated as a matrix of size $(M+M_\mathfrak{N})$ x $(M+M_\mathfrak{N})$ and $\mathbf{u}(\mathscr{A}^\dagger)$ and $\mathbf{f}(\mathscr{A}^\dagger)$ are column vectors both of length $M+M_\mathfrak{N}$.

3.3 Mathematical Properties of RBFs

The mathematical properties of differentiability and positive definiteness may be used to distinguish between the various RBF types and predict their behaviour. For instance, the class of globally-supported, infinitely-differentiable (i.e. C^∞) RBFs generally have higher error convergence rates than the globally-supported, finitely-differentiable and locally-supported RBF classes[8]. However, the C^∞ RBFs necessitate the inclusion of a user-selected shape parameter, ϵ, which controls how "peaked" the RBF is (e.g. for a Gaussian RBF, a large shape factor results in a sharp peak). The accuracy of the interpolant is strongly dependent on ϵ and proper selection is an active area of research. In general, the shape factor is permitted to vary in magnitude over the set of data centers (anisotropy). However, in practice the shape factor is typically specified to be uniform over the set of data centers (isotropy) for the sake of simplicity.

For a set of data centers $\mathscr{A}$ in m-dimensional Euclidean space, a Gaussian kernel of the form

$$\phi(\mathbf{r}, \boldsymbol{\alpha}) = \exp[-\epsilon^2(||\mathbf{r} - \boldsymbol{\alpha}||_2)^2], \quad \mathbf{r} \in \mathbb{R}^m, \boldsymbol{\alpha} \in \mathscr{A} \tag{75}$$

will be used throughout Chapters 5 and 6, where $||\mathbf{r} - \boldsymbol{\alpha}||_2$ represents the Euclidean distance between $\mathbf{r}$ and $\boldsymbol{\alpha}$. The choice of a Gaussian kernel may be justified by its mathematical properties:

- Symbolic differentiation of ϕ is simple, which is useful for the RBF discretization of operators that involve higher-order derivatives.
- C^∞ differentiability.

[8]The RBF literature, such as [37] and [38], cites spectral error convergence rates for the globally-supported, C^∞ RBFs. Verification of this claim is beyond the scope of this thesis.

- Positive definiteness which produces an invertible collocation matrix for uniquely chosen data centers [5][17][18][38].

In the computation of the RBK weight factors and polynomial coefficients, such as in Equations (55), (59), and (62), the inverse of the collocation matrix is never explicitly computed in practice. Rather, calling the built-in MATLAB® matrix division function [39] uses Gaussian elimination to solve for $\mathbf{w}$ and $\boldsymbol{\lambda}$.

Regardless of the positive definiteness of the Gaussian kernel, ill-conditioning can play an essential role in the accuracy of the interpolant. Specifically, as $||\boldsymbol{\alpha}_i - \boldsymbol{\alpha}_j||_2$ or ϵ approach zero the collocation matrix will become near-singular. For a distinct set of test centers with $||\boldsymbol{\alpha}_i - \boldsymbol{\alpha}_j||_2 \neq 0,\ \forall i, j$, this ill-conditioning arises from small shape factors (i.e. a "flat" Gaussian). However, interpolant accuracy also diminishes for large shape factors, wherein the Gaussian takes on the appearance of a sharp spike. Therefore, it would appear that interpolant accuracy and a well-conditioned collocation matrix are mutually exclusive; a phenomenon referred to as the "uncertainty principle" in the RBF literature [17][28].

One means for dealing with small-ϵ ill-conditioning is to use a "stable basis" derived from the standard-form RBK bases used in this chapter. In practice, these stable bases are formed via QR decomposition [19][20][21] or HSSVD [18][22] of ϕ. However, due to their mathematical complexity these stable methods will not be further pursued in this thesis. Rather, for the analyses in Chapters 5 and 6, ϵ will be selected naively by iterating until the discrepancy between the numerical and analytical solutions is reasonably small.

Chapter 4

Test Problem: LATIN Algorithm

In Section 4.1, the LATIN algorithm will be tested with a mixed FEM discretization of the spatial domain. The solution and its accuracy will be studied, including an analysis of reduced bases (i.e. discarding singular values and vectors) and the initial guess for the spacetime temperature field. In Section 4.2, these tests will be partially repeated with RBF discretization of the spatial domain and the results will be compared with FEM discretization.

General symbols:

Symbol	**Definition and (Units)**
$\square_1 \otimes \square_2$	tensor product of vectors or matrices $\square_1$ and $\square_2$
$\lvert \cdot \rvert$	absolute value
$\lvert\lvert \cdot \rvert\rvert_2$	Euclidean norm
$c_p = 1$	isotropic specific isobaric heat capacity of the domain ($\mathrm{Jkg^{-1}K^{-1}}$)
$d \in \mathbb{N}$	polynomial degree

$\widetilde{d} \in \mathbb{R}^+$	modified duality gap (J)
$\widetilde{d}_{eq} \in \mathbb{R}^+$	equilibrium duality gap (J)
div($\square$)	divergence operator on the field or vector $\square$
$k \in \mathbb{N}$	quantity of data centers used in the locally-supported RBFs
$K \in \mathbb{N}$	number of singular values and left/right singular vectors used in the ROM of $\mathbf{R}_2$
$l = 10$	length of the one-dimensional domain (m)
$m = 10$	quantity of free line segments in the domain
$\widetilde{m} = 1$	quantity of surfaces with prescribed Neumann BCs
$M = m + \widetilde{m}$	total quantity of free line segments and prescribed surfaces
$n = 10$	quantity of free nodes in the domain
$\widetilde{n} = 1$	quantity of nodes with prescribed Dirichlet BCs
$N = n + \widetilde{n}$	total quantity of nodes
$p \in \mathbb{N}$	iteration number of the LATIN algorithm
$q(x,t)$	flux field in continuous spacetime (Wm^{-2})
$\mathbf{R} \in \mathbb{R}^{n\mathrm{x}\tau}$	matrix of randomly selected numbers sampled from [-1,1]
$\mathbf{R}_2 \in \mathbb{R}^{n\mathrm{x}\tau}$	heat equation residuals in discrete spacetime (J)
$(\|\|\mathbf{R}_2\|\|_2)_{eq} \in \mathbb{R}^+$	heat equation residuals in the Euclidean norm at equilibrium (J)
$\widetilde{\mathbf{R}}_2 \in \mathbb{R}^{n\mathrm{x}\tau}$	ROM of $\mathbf{R}_2$ (J)

$s \in \mathbb{R}^+$	scaling factor (K)
$t \in (0, 10]$	time (s)
$u(x,t)$	temperature field in continuous spacetime (K)
$u_{SI}(x,t)$	semi-infinite temperature field in continuous spacetime (K)
$\widehat{\mathbf{U}}^0 \in \mathbb{R}^{n\text{x}\tau}$	initial guess for the KA temperature field in discrete spacetime (K)
$\mathbf{U}^* \in \mathbb{R}^{n\text{x}\tau}$	LATIN converged temperature field in discrete spacetime (K)
$\mathbf{U}_{SI} \in \mathbb{R}^{n\text{x}\tau}$	semi-infinite temperature field in discrete spacetime (K)
$x \in [0, 10]$	position (m)
$z = \binom{d+1}{d}$	number of monomials in the degree d polynomial

Greek symbols:

Symbol	**Definition and (Units)**
$\Delta t = 1$	uniform time-step size (s)
$\Delta x = 1$	uniform line segment length (m)
$\epsilon = 0.5$	RBF shape factor
$\kappa = 1$	isotropic thermal conductivity of the domain ($\text{Wm}^{-1}\text{K}^{-1}$)
$\rho = 1$	isotropic density of the domain (kgm^{-3})
$\tau = 10$	quantity of time-steps

4.1 Mixed FEM Discretization

The one-dimensional, transient thermal diffusion problem depicted in Figure 2 is to be solved via the LATIN algorithm illustrated in Section 2.4. For ease of reference, the governing equation and boundary/initial conditions are repeated

$$
\begin{aligned}
&-\operatorname{div}(q(x,t)) = \rho c_p \frac{\partial u}{\partial t},\ x \in (0,10)\ \text{m},\ t \in (0,10]\ \text{s} \\
&u(x=0,t) = u_{SI}(x=0,t),\ t \in (0,10]\ \text{s} \\
&q(x=l,t) = -1\ \text{Wm}^{-2},\ t \in (0,10]\ \text{s} \\
&u(x,t=0) = 0\ \text{K},\ x \in [0,10]\ \text{m}
\end{aligned}
$$

Where $u_{SI}(x=0,t)$ is the semi-infinite, time-varying temperature at $x = 0$ m; see Appendix C for a discussion of this boundary condition. $q(x=l,t) = -1$ Wm^{-2} is the Neumann thermal flux prescribed at $l = 10$ m and $u(x,t=0) = 0$ K is the uniform initial domain temperature. The thermal conductivity, density, and specific heat capacity are set at $\kappa = 1$ Wm^{-1}K^{-1}, $\rho = 1$ kgm^{-3}, and $c_p = 1$ Jkg^{-1}K^{-1}, respectively.

To begin, the spatial domain is discretized into $N = 11$ equispaced nodes: $\widetilde{n} = 1$ Dirichlet boundary node at $x_1 = 0$ m and $n = 10$ free nodes at $x_2 = 1$ m, $\cdots, x_{11} = 10$ m. The nodes are connected by $M = 11$ line segments of length $\Delta x = 1$ m: $m = 10$ free line segments connecting x_1 to $x_2, \cdots, x_{10}$ to x_{11} and $\widetilde{m} = 1$ Neumann boundary surface at $x_{11} = 10$ m. Recall from Section 2.4 that the line elements comprise the primal grid whereas the nodes make up the dual grid.

In combination with the discrete space domain, a backward Euler scheme is used to discretize the time domain into $\tau = 10$ time-steps of length $\Delta t = 1$ s, starting at $t = 0$ s and ending at $t = 10$ s.

It is assumed that the space component of $u(x,t)$ is spanned by a set of $N = 11$ one-dimensional, piecewise linear "hat functions". Likewise, it is assumed that the space component of $q(x,t)$ is spanned by a set of $m = 10$ one-dimensional, piecewise (discontinuous) constant functions. From these basis functions, the left-hand side matrices of SLE (24) may be computed; see Appendix D.1 for sample calculations.

An initial guess $\widehat{\mathbf{U}}^0$ for the KA spacetime temperature field is required to initialize the LATIN algorithm. This initial guess will be generated randomly, like so

$$\widehat{\mathbf{U}}^0 = s\mathbf{R} \tag{76}$$

where $\mathbf{R}$ is a 10 x 10 matrix of randomly generated numbers sampled from [-1,1] and s is a scaling factor. A range of scaling factors will later be used to test the robustness of the LATIN algorithm, however a factor of $s = 0.5$ will be used until specified otherwise.

Convergence of the algorithm is controlled by the duality gap, as computed by Equation (49). A threshold duality gap of 1E$-$3 is used with a failsafe of fifty maximum iterations in case the algorithm fails to converge. With these criteria, the LATIN algorithm outlined on page 50, and the input data defined above, the following converged spacetime temperature field is computed:

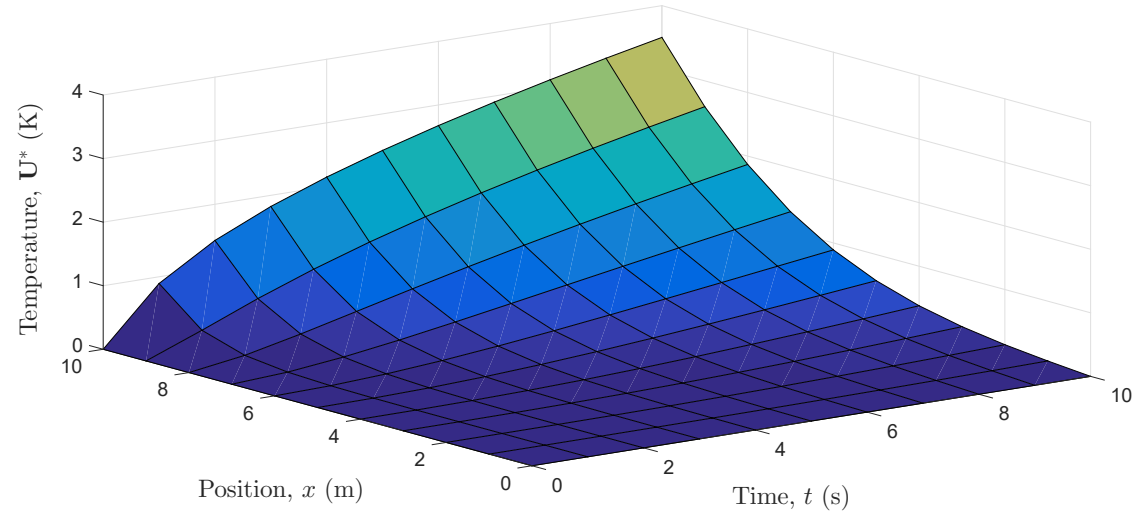


Figure 3: LATIN converged temperature as a function of position and time. A reduced basis dimension of $K = 10$ was used (i.e. none of the singular values were discarded).

The error of the converged LATIN temperature field is estimated with respect to the semi-infinite solution in Equation (116):

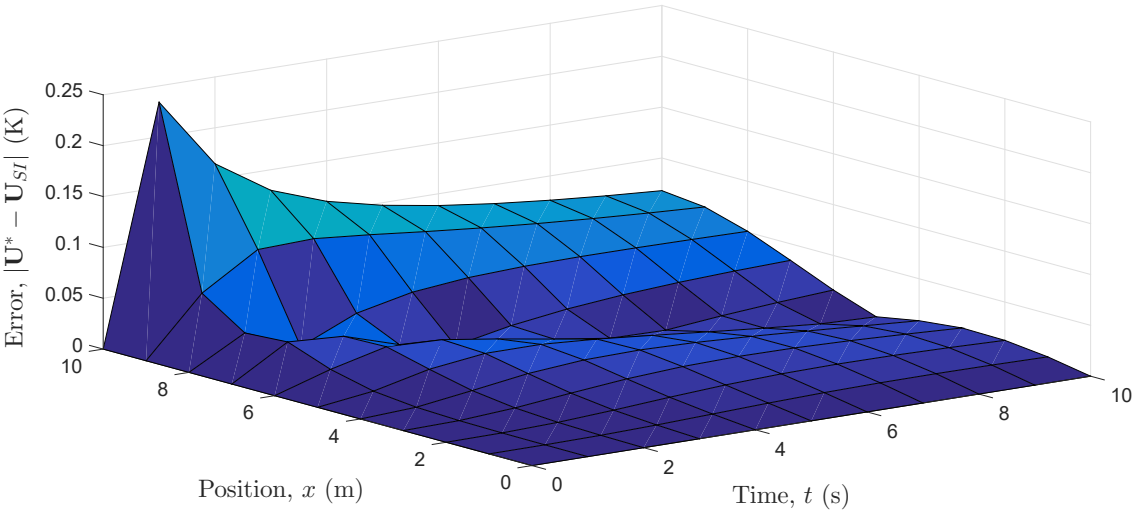


Figure 4: Error between the LATIN converged and semi-infinite temperatures as a function of position and time. The error between the two solutions in the Euclidean norm is 0.467.

Figure 4 elicits an error field that is largest near $x = 10$ m, peaking at $t = 1$ s and then decreasing in time. This result would suggest that the Neumann flux, which is instantaneously applied at $(x, t) = (10,1)$, imposes a thermal shock on the system. This effect is initially localized at the right end of the domain, however as time progresses the flux propagates towards the left end of the domain with an attendant decrease in error.

Recall from Subsection 2.4.3 that the heat equation residual matrix, $\mathbf{R}_2$, is decomposed with SVD to form a reduced basis. The $\tau - K$ smallest singular values and associated left and right singular vectors of $\mathbf{R}_2$ are discarded to form the reduced order $\widetilde{\mathbf{R}}_2$ (alternatively, the K largest singular values and associated left and right singular vectors are kept). To study the effect of the discarded singular values on the LATIN solution, the error is plotted against time while varying K. To avoid overcrowding of the graph, only the errors at $x = 10$ m will be plotted as they are generally of greatest magnitude at that location:

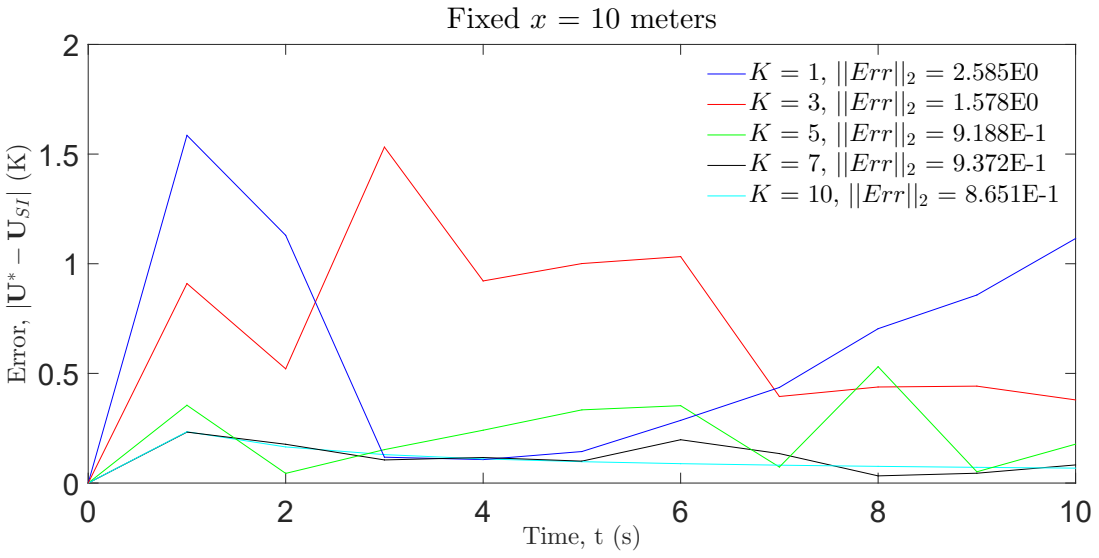


Figure 5: Error between the LATIN converged and semi-infinite temperatures at $x = 10$ m and $t = 0,...,10$ s for $K = (1,3,5,7,10)$. Also shown are the Euclidean norm errors for each K.

Figure 5 shows a wavelike error between the LATIN and semi-infinite temperatures. The amplitudes of these error waves appear to grow in magnitude as $\widetilde{\mathbf{R}}_2$ is spanned with fewer basis vectors (i.e. smaller K). Conversely, $K = \tau = 10$ results in a comparatively smooth error as a function of time. However, larger K incur greater computational expense in calculating the temperature corrections for each LATIN iteration. Although this trend is negligible for such a small test problem, larger problems may require a careful balance between the solution accuracy and performance of the algorithm.

To assess the convergence characteristics of the LATIN algorithm, the duality gap from Equation (49) is plotted below against the iteration number for the same range of K used above:

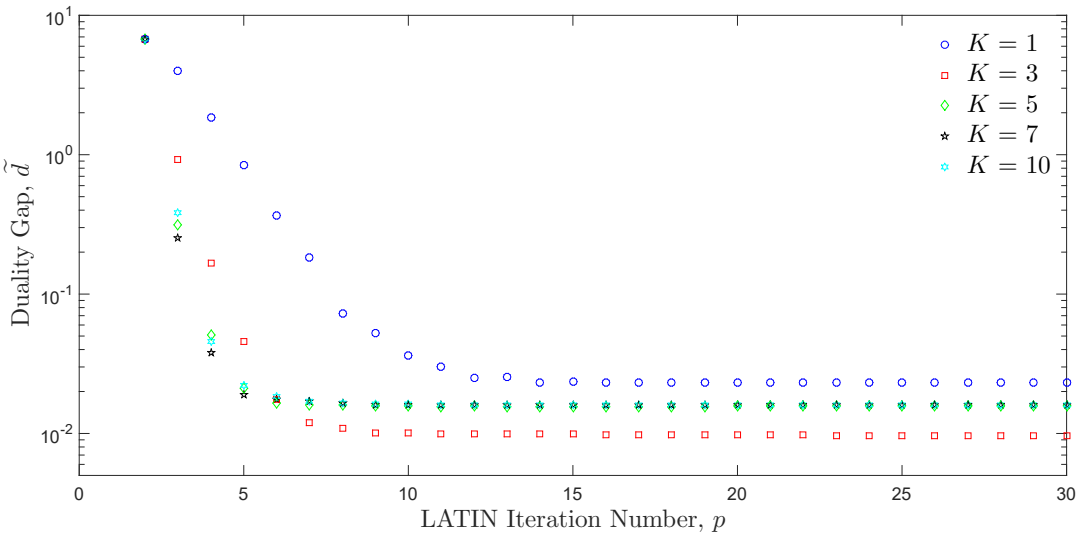


Figure 6: Semi-log plot of the duality gap magnitude versus the iteration number of the LATIN algorithm for K = (1,3,5,7,10).

Figure 6 suggests that the duality gap decreases monotonically, converging to an equilibrium in approximately 15 iterations. These equilibrium values, labelled as $\widetilde{d}_{eq}$, are summarized in Table 1 for each K.

The heat equation residuals give another useful measure of the solution accuracy and are plotted below against the LATIN iteration number:

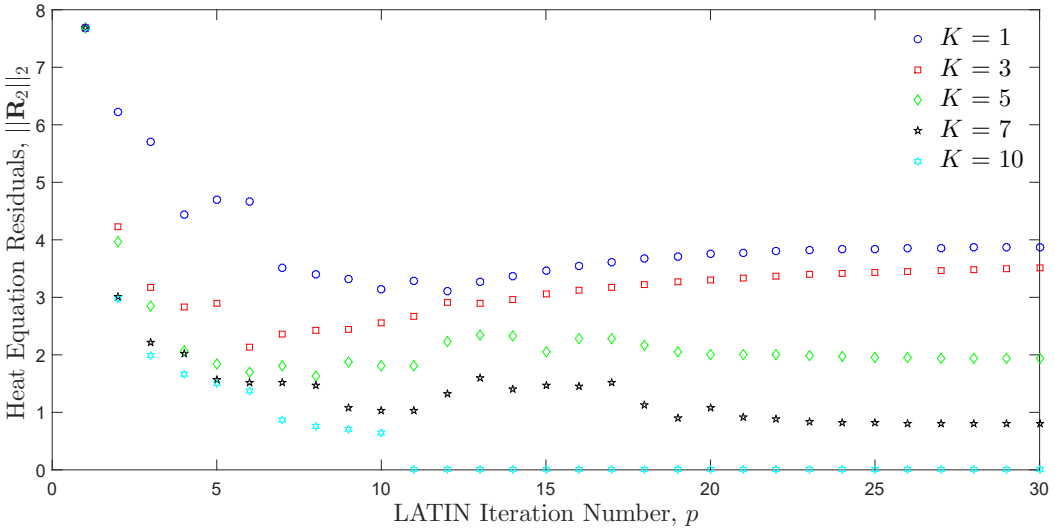


Figure 7: Discrete heat equation residuals (Euclidean norm) versus the iteration number of the LATIN algorithm for K = (1,3,5,7,10).

Figure 7 indicates that the residuals converge to an equilibrium in approximately 25 iterations. The following table summarizes the equilibrium values of the duality gaps and residuals for each K:

Table 1: $\widetilde{d}_{eq}$ and $(||\mathbf{R}_2||_2)_{eq}$ for K = (1,3,5,7,10) and fixed $s = 0.5$.

	$\widetilde{d}_{eq}$	$(\|\|\mathbf{R}_2\|\|_2)_{eq}$
$K = 1$	2.307E-2	3.871E0
$K = 3$	9.671E-3	3.507E0
$K = 5$	1.572E-2	1.932E0
$K = 7$	1.606E-2	8.013E-1
$K = 10$	1.609E-2	2.128E-15

Table 1 shows an unpredictable trend; intuition would suggest that the higher K should give smaller $\widetilde{d}_{eq}$, however $K = 3$ gives a minimum of $\widetilde{d}_{eq} = 9.671\text{E-}3$. The cause of this seemingly conflicting outcome is not known. The right-most column of

Table 1 conveys a more predictable trend, wherein the smaller $(||\mathbf{R}_2||_2)_{eq}$ occur at larger K. It is not known why $(||\mathbf{R}_2||_2)_{eq}$ decreases so steeply from $K = 7$ to $K = 10$.

As was mentioned prior, the robustness of the LATIN algorithm may be tested by varying the scaling factor in Equation (76). The duality gap and residuals are plotted below against the LATIN iteration number for a range of s:

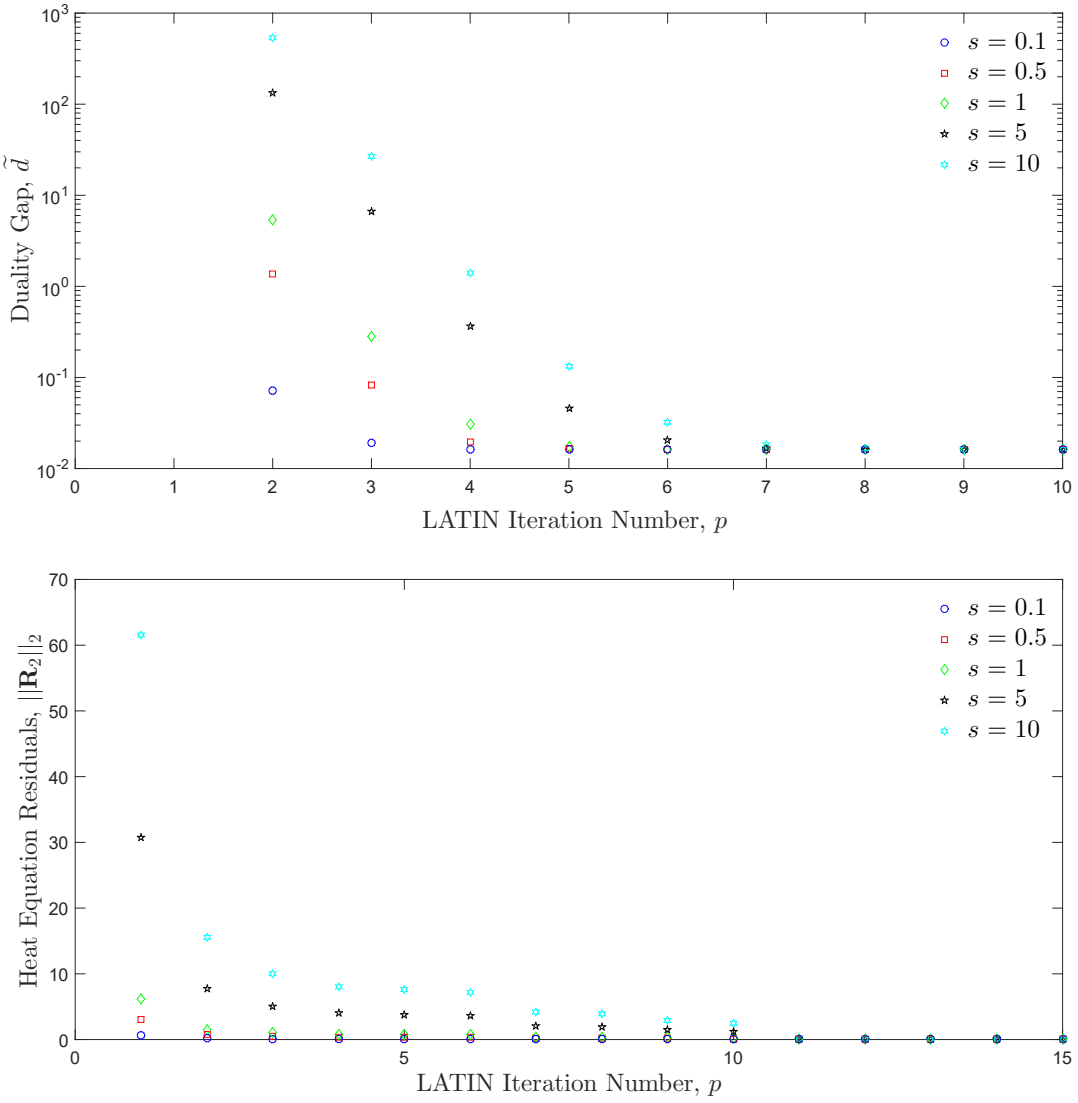


Figure 8: Duality gap (top, semi-log) and Euclidean norm heat equation residuals (bottom) versus the iteration number of the LATIN algorithm for s = (0.1,0.5,1,5,10). A reduced-basis dimension of $K = 10$ was used to generate all of the data above.

Figure 8 collectively suggests that all duality gaps (resp. residuals) successfully converge at approximately iteration number seven (resp. eleven). Further, it may be observed that the rates of convergence depend intuitively on the perturbation scaling factor. Specifically, the highly-perturbed (i.e. large s) initial guesses converge slower than the lowly-perturbed guesses. This makes mathematical sense as the highly-perturbed guesses are initially highly inaccurate (as reflected by the duality gaps at iteration two and residuals at iteration one). Thus, the LATIN algorithm must apply larger temperature corrections to the highly-perturbed solutions in order to converge.

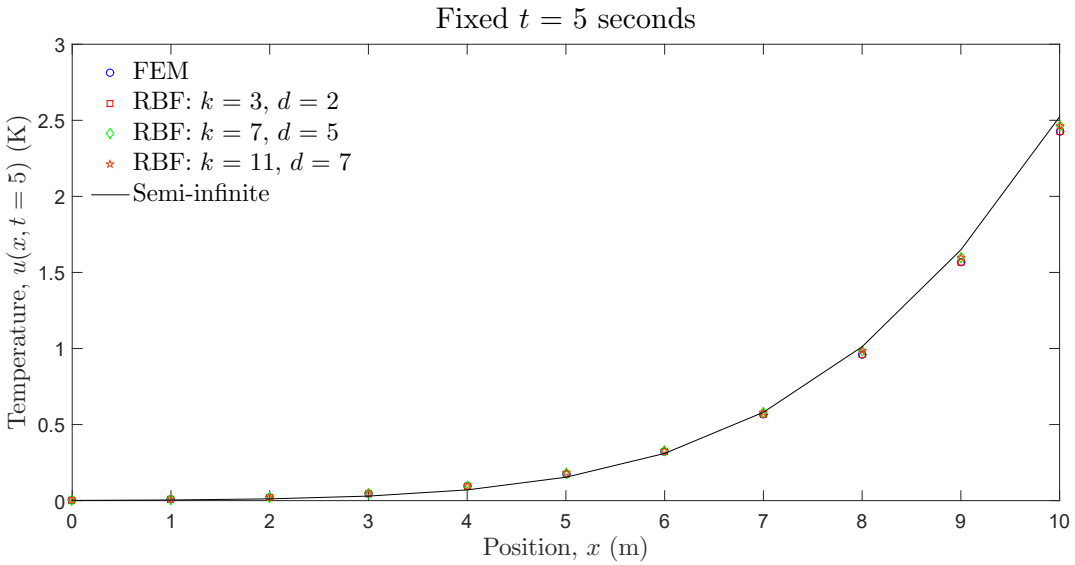
4.2 RBF Discretization

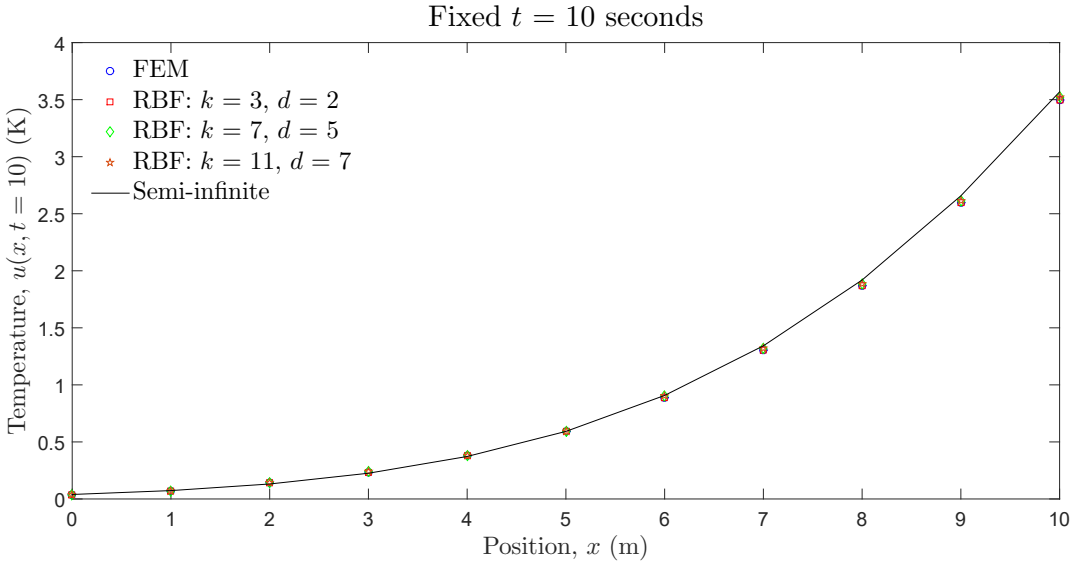
The LATIN analysis from Section 4.1 is partially repeated now with an RBF approach in place of FEM. The algorithm itself remains largely unchanged with the following exceptions: 1) the $m = 10$ primal line segments are replaced with data centers located at the line segment centroids 2) the basis functions for the space components of $u(x,t)$ and $q(x,t)$ are replaced with Gaussian RBKs per Equation (75). Sample calculations for the RBF discretization scheme may be found in Appendix D.2.

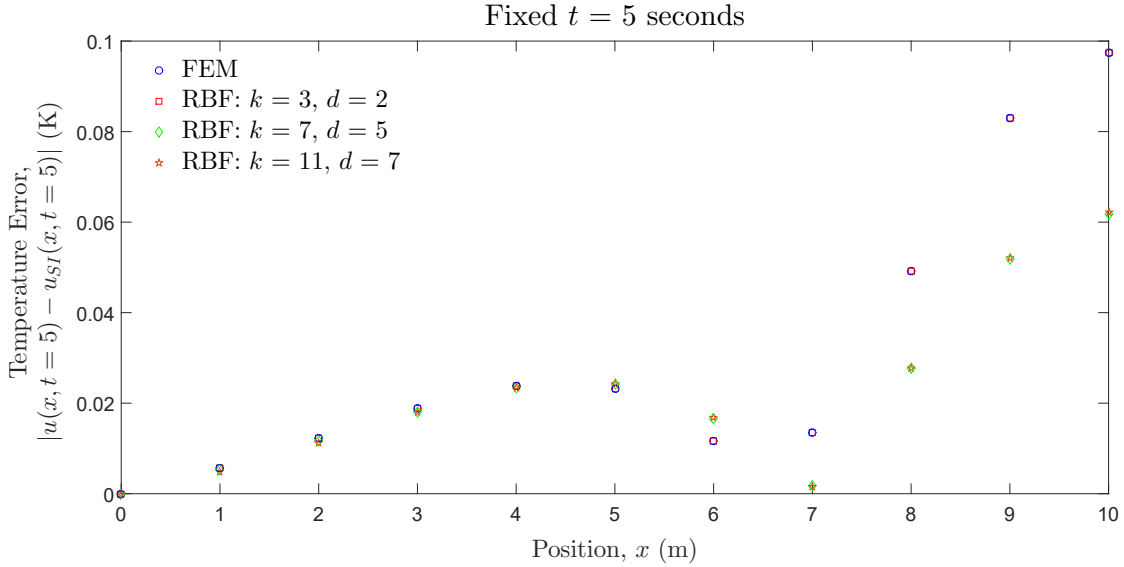
To study the effects of stencil size and polynomial order, the following RBF parameters will be tested:

- Stencil size $k = 3$, polynomial degree $d = 2$, shape factor $\epsilon = 0.5$.
- Stencil size $k = 7$, polynomial degree $d = 5$, shape factor $\epsilon = 0.5$.
- Stencil size $k = 11$, polynomial degree $d = 7$, shape factor $\epsilon = 0.5$.

Using the same geometry, boundary conditions, and thermal properties as in Section 4.1 enables comparison between the FEM and RBF solutions. Additionally, the LATIN parameters are fixed at $K = 10$ and $s = 0.5$:


Fixed $t = 5$ seconds
FEM
RBF: $k = 3$, $d = 2$
RBF: $k = 7$, $d = 5$
RBF: $k = 11$, $d = 7$
Semi-infinite
Temperature, $u(x, t = 5)$ (K)
Position, x (m)


Fixed $t = 10$ seconds
FEM
RBF: $k = 3$, $d = 2$
RBF: $k = 7$, $d = 5$
RBF: $k = 11$, $d = 7$
Semi-infinite
Temperature, $u(x, t = 10)$ (K)
Position, x (m)


Fixed $t = 5$ seconds
FEM
RBF: $k = 3$, $d = 2$
RBF: $k = 7$, $d = 5$
RBF: $k = 11$, $d = 7$
Temperature Error, $|u(x, t = 5) - u_{SI}(x, t = 5)|$ (K)
Position, x (m)

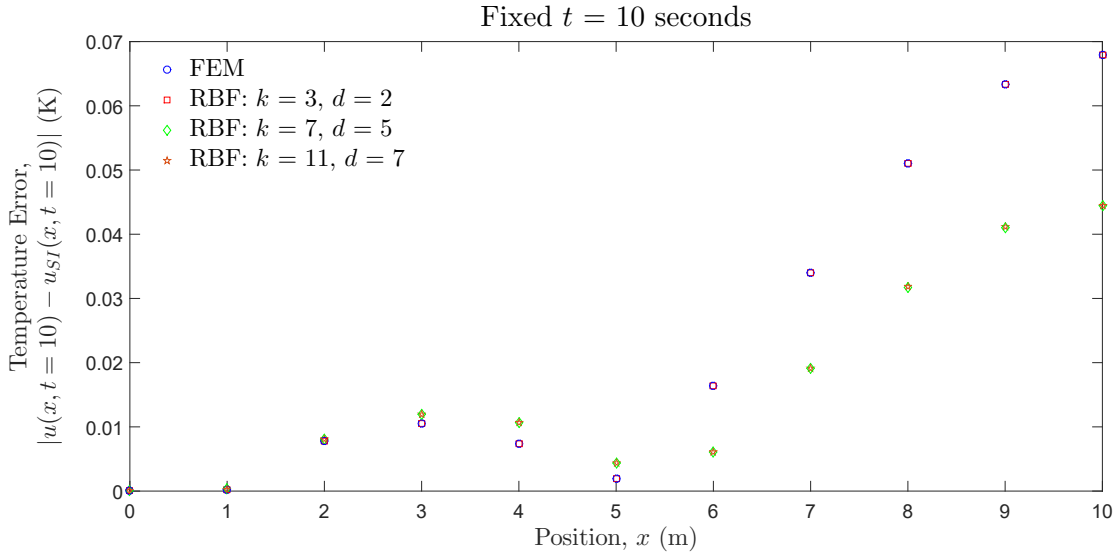


Figure 9: Temperature and error versus position at times of t = 5 and 10 seconds using FEM and RBF discretization for the numerical solutions.

Table 2: Euclidean norm of the error between the FEM/RBF solutions and the semi-infinite solution at times of $t = 5$ and 10 seconds.

	$\|\|u(x,t) - u_{SI}(x,t)\|\|_2$	
	$t = 5$ s	$t = 10$ s
FEM	1.442E-1	1.136E-1
RBF, $k = 3$, $d = 2$	1.442E-1	1.136E-1
RBF, $k = 7$, $d = 5$	9.579E-2	7.351E-2
RBF, $k = 11$, $d = 7$	9.613E-2	7.635E-2

Figure 9 and Table 2 above reveal a nearly identical accuracy in the RBFs with $(k, d) = (7, 5)$ and $(11, 7)$, both of which are superior to the FEM formulation. This difference in performance likely comes from the mathematical properties of the RBFs, which span the temperature field with a basis of infinitely-differentiable Gaussian functions and a polynomial. Moreover, each of these basis functions are supported over stencils consisting of k RBF data centers. By contrast, the FEM spans the temperature field with a basis of once-differentiable hat functions, each of which

is supported only over two adjacent line segments. The local support and higher differentiability of the RBFs account for the contrasting trends seen in Figure 9.

However, the larger stencils used with the RBFs can also incur substantial computing costs. In particular, the Gramian matrices which make up the left-most block matrix in SLE (24) each have bandwidths of k. Those same Gramian matrices in the FEM formulation are comparatively sparse with bandwidths of one or two. Furthermore, the number of terms comprising the polynomial grows rapidly with degree as $z = \binom{d+1}{d}$. Each of these z terms must be evaluated at all $N + M$ data centers in the primal and dual grids.

Chapter 5

Test Problems: Scattered Data Interpolation with RBFs

The application of RBFs to scattered data interpolation will be introduced by way of two test problems: a two-dimensional velocity field in a rheometer and a two-dimensional stress field around a circular hole in a semi-infinite plate. These two test cases will illustrate interpolation of complex data sets, respectively velocity vector fields and stress tensor fields.

General symbols:

Symbol	**Definition and (Units)**
$\lvert \cdot \rvert$	absolute value
$\lvert\lvert \cdot \rvert\rvert_p$	Euclidean ($p = 2$) or infinity ($p = \infty$) norm
$a = 0.5$	hole radius (m)
$d \in \mathbb{N}$	polynomial degree
$\mathrm{div}(\square)$	divergence of the vector or field $\square$

$Err \in \mathbb{R}$	relative error of the stress or stress gradient tensor
$f \in \mathbb{N}$	frequency with which the RBF data are sampled from the SPH data
$k \in \mathbb{N}$	quantity of data centers used in the locally-supported RBFs
$N = 170$	number of RBF data centers and derivative centers used to discretize a quadrant of the semi-infinite plate
$r \in \mathbb{R}^+$	radius measured from the center of the hole (m)
$R = 0.035$	singularity clustering parameter (m)
$u \in \mathbb{R}$	velocity in the x-dimension (ms^{-1})
$v \in \mathbb{R}$	velocity in the y-dimension (ms^{-1})
$V_{Int} \in \mathbb{R}^+$	interpolated velocity magnitude (ms^{-1})
$V_{SPH} \in \mathbb{R}^+$	SPH velocity magnitude (ms^{-1})
$x \in \mathbb{R}$	position in the x-dimension (m)
$y \in \mathbb{R}$	position in the y-dimension (m)

Greek symbols:

Symbol	**Definition and (Units)**
$\epsilon \in \mathbb{R}^+$	RBF shape factor
$\omega = 20$	angular velocity of the rheometer cylinder (s^{-1})
$\phi = 1.5$	outer radius of the rheometer cylinder (cm)

$\sigma_A = 10$	stress applied parallel to the x-axis at the ends of the semi-infinite plate (MPa)
$\sigma_\square \in \mathbb{R}$	component $\square$ of the stress tensor (Nm^{-2})
$\dfrac{\partial \sigma_\square}{\partial r} \in \mathbb{R}$	component $\square$ of the radial stress gradient tensor (Nm^{-3})
$\dfrac{\partial \sigma_\square}{\partial \theta} \in \mathbb{R}$	component $\square$ of the circumferential stress gradient tensor (Nm^{-3})
$\theta \in \mathbb{R}$	angle measured from the positive x-axis (radians)

The subscripts below will be used to indicate particle indices and components of the stress or stress gradient tensors:

Symbol	**Definition**
$*$	analytical stress or stress gradient
i	particle/RBF center index
rr	radial component
$\theta\theta$	circumferential component
$r\theta$	shear component

5.1 Two-Dimensional Velocity Field in a Rheometer

The standard RBF interpolation framework will be introduced with a steady-state, two-dimensional velocity field in a rheometer:

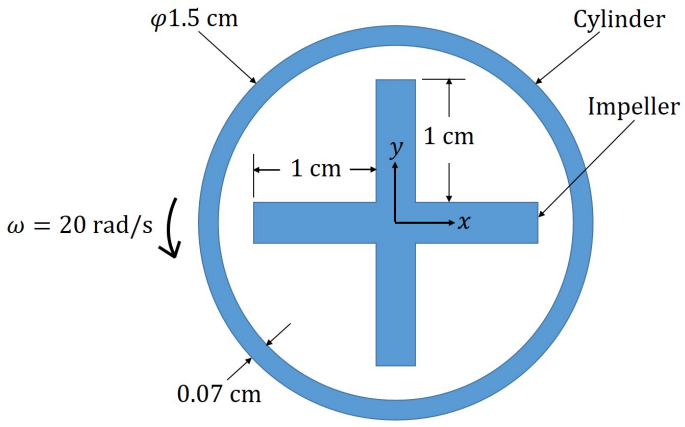


Figure 10: Schematic of the two-dimensional rheometer. The impeller is held stationary while the cylinder is rotated counter-clockwise at $\omega = 20$ rad/s.

The velocity data was generated using a Smoothed Particle Hydrodynamics (SPH) simulation with 2,788 particles and 1,000 uniform time-steps of 0.002 seconds. Steady-state was assumed to occur at time-step 200, wherein the locations (x, y) and velocities (u, v) of the 2,788 particles were stored. The RBF data centers and velocities will be extracted from this particle data with a user-controlled frequency of f. For instance, $f = 5$ would extract the following RBF data from the SPH data:

SPH Data (2,788 data points)		RBF Data (559 data points)
(x_1, y_1, u_1, v_1)		(x_1, y_1, u_1, v_1)
(x_2, y_2, u_2, v_2)		(x_6, y_6, u_6, v_6)
(x_3, y_3, u_3, v_3)		$(x_{11}, y_{11}, u_{11}, v_{11})$
$\vdots$	$f = 5 \Rightarrow$	$\vdots$
$(x_{2,786}, y_{2,786}, u_{2,786}, v_{2,786})$		$(x_{2,781}, y_{2,781}, u_{2,781}, v_{2,781})$
$(x_{2,787}, y_{2,787}, u_{2,787}, v_{2,787})$		$(x_{2,786}, y_{2,786}, u_{2,786}, v_{2,786})$
$(x_{2,788}, y_{2,788}, u_{2,788}, v_{2,788})$		$(x_{2,788}, y_{2,788}, u_{2,788}, v_{2,788})$

Using the 559 RBF data points above as input, the velocities will be interpolated back on to the 2,788 SPH particles. The effects of the data sampling frequency f and polynomial degree d on the interpolant accuracy will also be studied. Each of these tests will use globally-supported (i.e. a stencil size of $k = 559$ for the example above), standard RBF interpolation. For the time being, a Gaussian RBK with an experimentally determined shape factor of $\epsilon = 1.5\text{E}{+3}$ will be used with $f = d = 5$. This parameter set produces the following velocity vector plots:

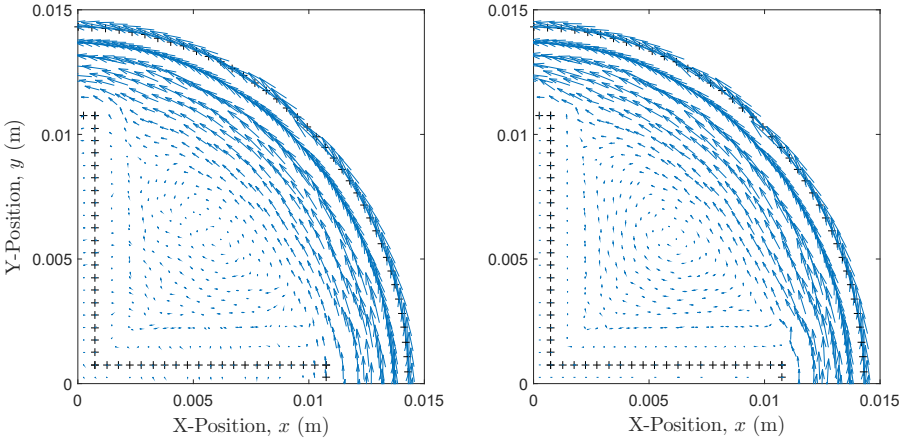


Figure 11: Vector plots of the interpolated (left) and SPH (right) velocity fields in the top-right rheometer quadrant. The vector arrow lengths are scaled by a factor of 3:1 cm/s. The black "+" symbols indicate the solid boundaries of the impeller and cylinder.

Figure 11 conveys good agreement between the interpolated and SPH fields near the cylindrical boundary. However, the vortex seen in the right plot appears to be "smeared out" in the left plot. Also apparent from the left plot are the physically impossible, non-zero velocities on and inside the boundary of the impeller. These two trends suggest that the interpolant is inducing a small, yet spurious velocity field in the vortex and impeller regions where the SPH velocities are of smaller or zero magnitude. The effect of the parameters d and f on this anomaly will be studied later. To more clearly see the discrepancies between the interpolated and SPH velocity fields, the difference in velocity magnitudes $|V_{Int,i} - V_{SPH,i}|$ of particles $i = 1, ..., 2,788$ is plotted in Figure 12:

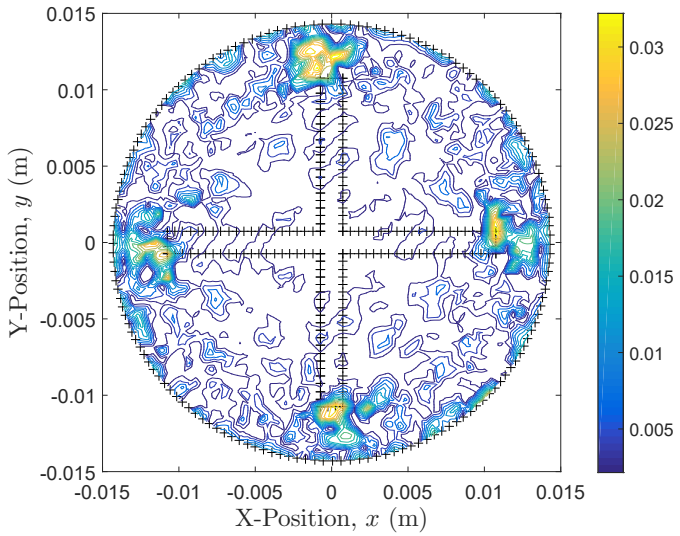


Figure 12: Contour plot of the differences between the interpolated and SPH velocity magnitudes.

Evidently, the maximal differences occur in the four regions between the impeller tips and cylindrical boundary. In these regions, the flow accelerates as it passes through the constriction and then decelerates as it exits, resulting in a complex flow pattern. It is plausible that the difference is largest in these regions where the velocity is rapidly changing in both magnitude and direction. Away from these highly localized areas of large difference, the discrepancies between V_{Int} and V_{SPH} appear to be of smaller

magnitude and are sporadically located throughout the rheometer domain.

The effects of the data sampling frequency and polynomial degree on the interpolant accuracy will now be studied via the following plots:

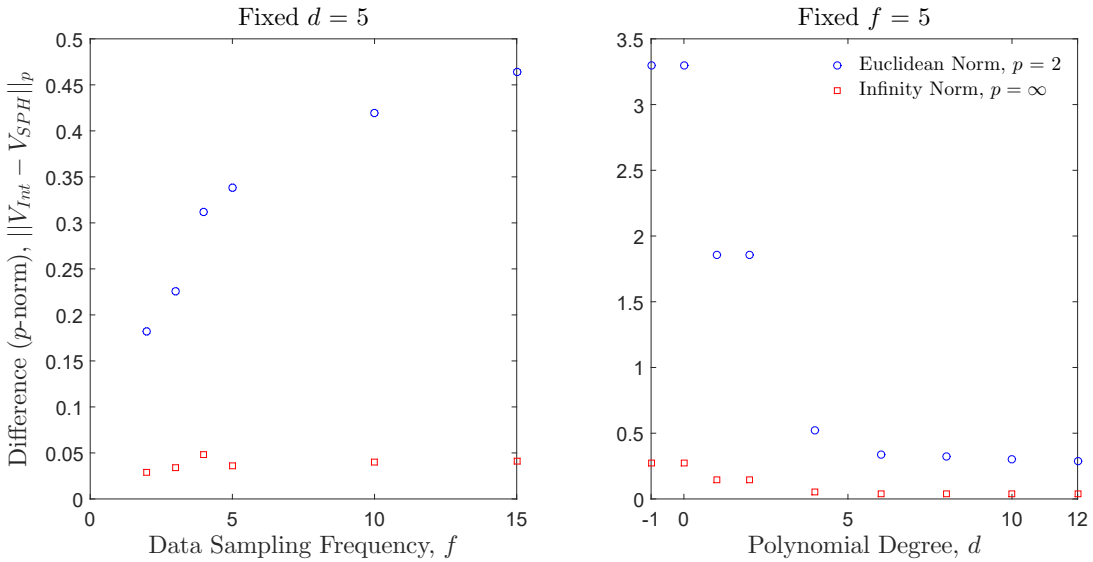


Figure 13: Difference in the Euclidean and infinity norms versus the data sampling frequency (left) and polynomial degree (right). A polynomial degree of -1 indicates that no polynomial basis is used (i.e. purely RBF).

The left plot of Figure 13 intuitively elicits an inverse proportionality between the difference and sampling frequency. In other words, using more of the velocity data, and therefore more Gaussian kernels, increases the accuracy of the resulting interpolant. The right plot also suggests an inverse proportionality, this time between the difference and degree of the polynomial basis. However, it can also be seen that a trend of diminishing returns governs for large d, wherein the difference decreases at a slower rate than for small d. In addition, large f and/or d incur substantial costs when storing the necessary vectors and matrices. Collectively, these observations justify the use of a more modest combination of f and d, which were shown in Figures 11 and 12 to yield good agreement with the SPH velocity field.

For the sake of argument, however, the vector plot in Figure 11 will be reproduced using a high polynomial degree in an effort to more accurately interpolate the vortex:

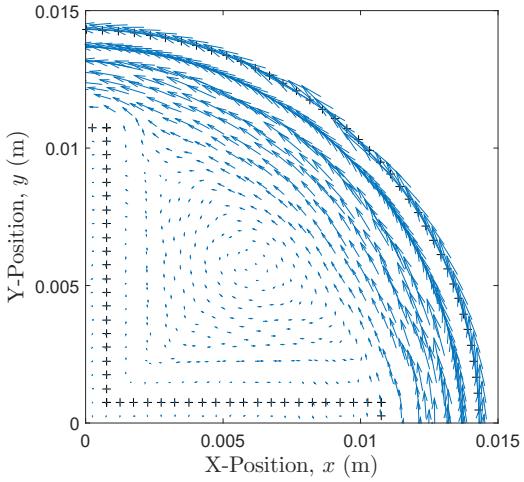


Figure 14: Vector plot of the interpolated velocity field in the top-right quadrant of the rheometer for $f = 5$ and $d = 12$.

It may be seen in Figure 14 that the degree 12 polynomial more accurately captures the vortex relative to $d = 5$. This result suggests that a higher-d polynomial basis may be necessary to properly interpolate intricate flow structures. However, lower-d polynomials seem to be sufficient for interpolating the bulk flows in the remainder of the rheometer domain.

5.2 Two-Dimensional Stress and Stress Gradient Fields in a Semi-Infinite Plate with a Circular Hole

As a test of the standard and Hermite RBF interpolation schemes, both methods will be used to reconstruct the two-dimensional stress and stress gradient fields for a simple geometry. Specifically, a semi-infinite plate with a circular hole is subjected to uniaxial tension. Symmetry of the resulting stress field allows for the analysis of a single quadrant of the plate, as illustrated in the diagram below:

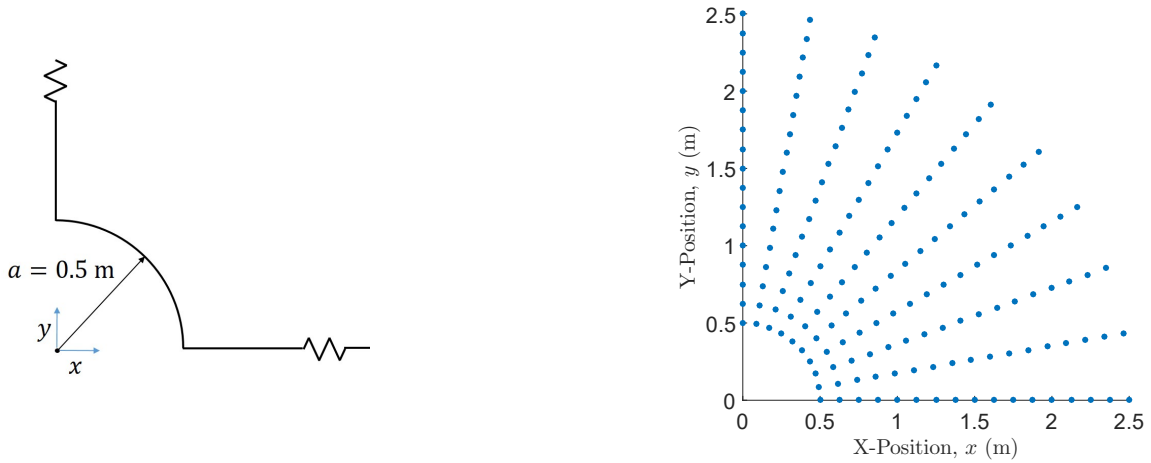


Figure 15: A semi-infinite plate with a hole radius of $a = 0.5$ m and applied tensile stress of $\sigma_A = 10$ MPa (left) and $N = 170$ coincident data and derivative centers (right).

The geometry in Figure 15 appears well-disposed to polar coordinates; indeed, the Kirsch equations (see below) are exclusively expressed in this way in the literature. However, it turns out that the theoretical framework presented in Subsections 3.1.1 and 3.1.2 is most naturally expressed in Cartesian coordinates. Therefore, the interpolation algorithms used with this test problem follow Subsections 3.1.1 and 3.1.2 closely and conversion to polar coordinates is only done in the post-processing step.

The Kirsch equations [40] satisfy the quasi-static conservation of momentum equation $\mathrm{div}(\boldsymbol{\sigma}) = 0$ and represent an exact solution of the problem introduced in Figure 15. They will hereafter be superscripted with a star to distinguish them from the interpolated stress field

$$\begin{aligned}
\sigma^*_{rr} &= \frac{\sigma_A}{2}\left[\left(1-\frac{a^2}{r^2}\right)+\left(1+3\frac{a^4}{r^4}-4\frac{a^2}{r^2}\right)\cos(2\theta)\right] \\
\sigma^*_{\theta\theta} &= \frac{\sigma_A}{2}\left[\left(1+\frac{a^2}{r^2}\right)-\left(1+3\frac{a^4}{r^4}\right)\cos(2\theta)\right] \\
\sigma^*_{r\theta} &= -\frac{\sigma_A}{2}\left[\left(1-3\frac{a^4}{r^4}+2\frac{a^2}{r^2}\right)\sin(2\theta)\right]
\end{aligned} \tag{77}$$

The equations above inherently assume that the applied stress is insufficient to cause plastic deformation of the plate. The exact stress gradient field may be expressed as

$$\begin{aligned}
\frac{\partial\sigma^*_{rr}}{\partial r} &= \frac{\sigma_A}{2}\left[\left(2\frac{a^2}{r^3}\right)-\left(8\frac{a^2}{r^3}-12\frac{a^4}{r^5}\right)\cos(2\theta)\right] \\
\frac{\partial\sigma^*_{\theta\theta}}{\partial\theta} &= \sigma_A\left[\left(1+3\frac{a^4}{r^4}\right)\sin(2\theta)\right] \\
\frac{\partial\sigma^*_{r\theta}}{\partial r} &= -\frac{\sigma_A}{2}\left[\left(12\frac{a^4}{r^5}-4\frac{a^2}{r^3}\right)\sin(2\theta)\right] \\
\frac{\partial\sigma^*_{r\theta}}{\partial\theta} &= -\sigma_A\left[\left(1-3\frac{a^4}{r^4}+2\frac{a^2}{r^2}\right)\cos(2\theta)\right]
\end{aligned} \tag{78}$$

Note that the $\partial\sigma_{rr}/\partial\theta$ and $\partial\sigma_{\theta\theta}/\partial r$ stress gradient components are not considered because they are not required to compute $\mathrm{div}(\boldsymbol{\sigma})$. The large gradients in the vicinity of the hole will be used to test the robustness of each interpolation scheme.

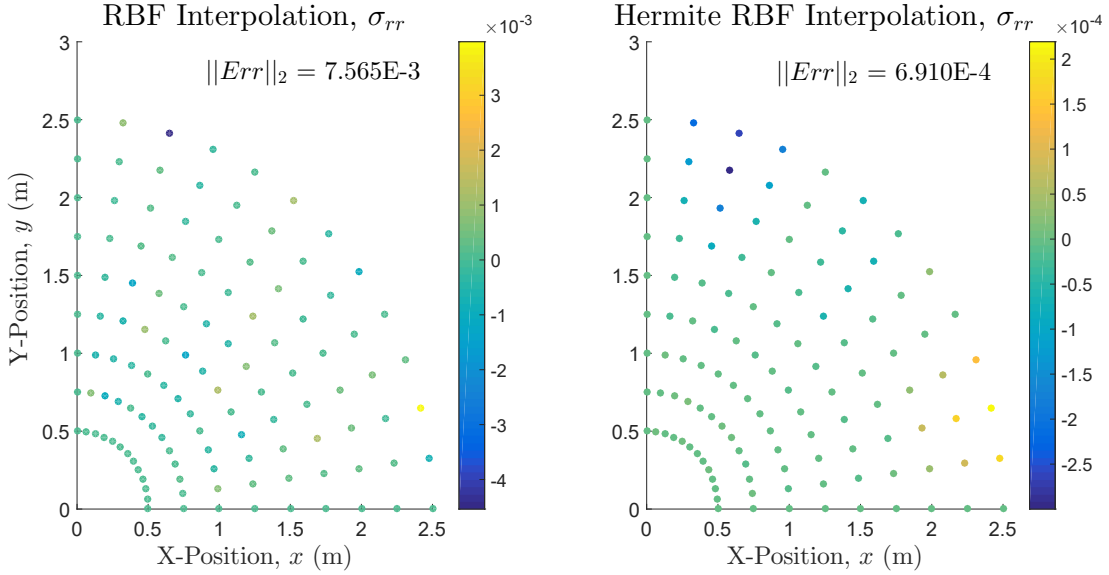
5.2.1 Stress and Stress Gradient Interpolation

In the standard RBF interpolation, the three stress components in (77) are evaluated at all $N = 170$ data centers depicted in Figure 15. This generates a stress tensor field which represents the given problem data. The Hermite RBF interpolation extends the

data field by invoking the gradients of the exact stresses. In this step, the four stress gradient components in (78) are evaluated at the 170 data centers and appended to the stress data field from (77).

As per Equation (75), a set of locally-supported Gaussian RBKs are used in conjunction with the data to assemble and solve the constraint Equations (55) and (59). The resultant weight factors are then used to approximate the stress fields at 117 interpolation centers. Identical inputs to the standard and Hermite interpolation algorithms include an experimentally determined shape factor of $\epsilon = 2$, polynomial degree of $d = 1$, and stencil size of $k = 15$.

The relative error between the interpolated and exact stress fields may be computed at interpolation centers $i = 1, ..., 117$ via $Err_i = \dfrac{\sigma_i - \sigma_i^*}{\sigma_A}$. The six plots below depict the interpolation centers by circles, with each circle coloured by the relative error in stress component $\sigma_{\square}$. Note that the plots at left titled "RBF Interpolation" refer to the standard RBF interpolation scheme:



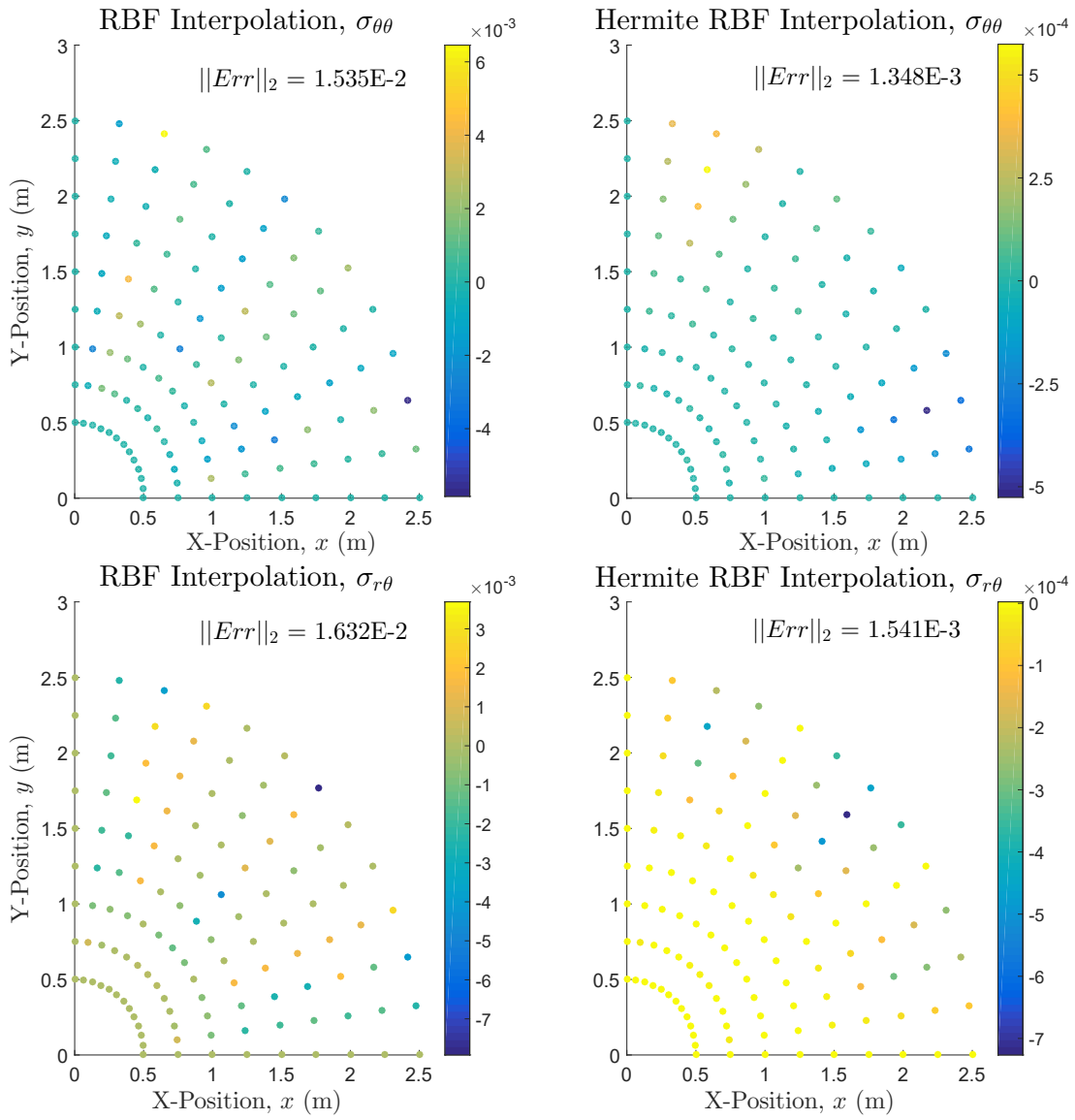
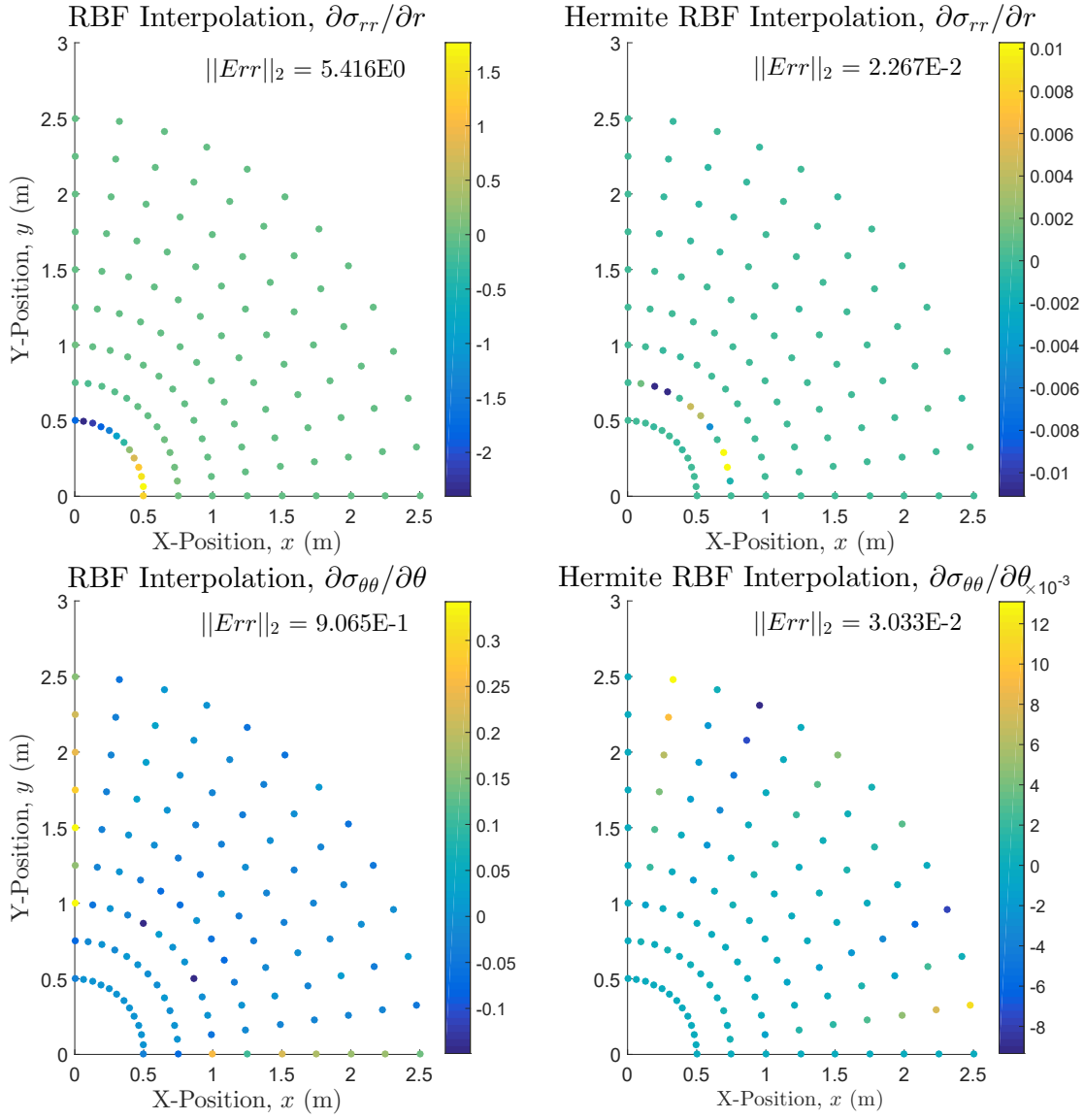


Figure 16: Interpolation centers coloured by the relative error of the radial, circumferential, and shear stress components. The Euclidean norm of the error is shown at the top right of each plot.

Evidently, the error fields computed with Hermite interpolation are an order of magnitude smaller than those of the standard interpolation. Further discussion of this trend will follow Figure 17.

The stress gradients may be approximated at the interpolation centers $i = 1, ..., 117$ using the technique introduced on page 72. The relative error between the interpolated and exact gradients may in turn be calculated as $Err_i = \frac{1}{\sigma_A}\left[\frac{\partial \sigma_i}{\partial \square} - \frac{\partial \sigma_i^*}{\partial \square}\right]$, where $\square = r$ refers to the radial gradient and $\square = \theta$ the circumferential gradient. The plots below depict the interpolation centers coloured by the relative error in each component of the stress gradient tensor:



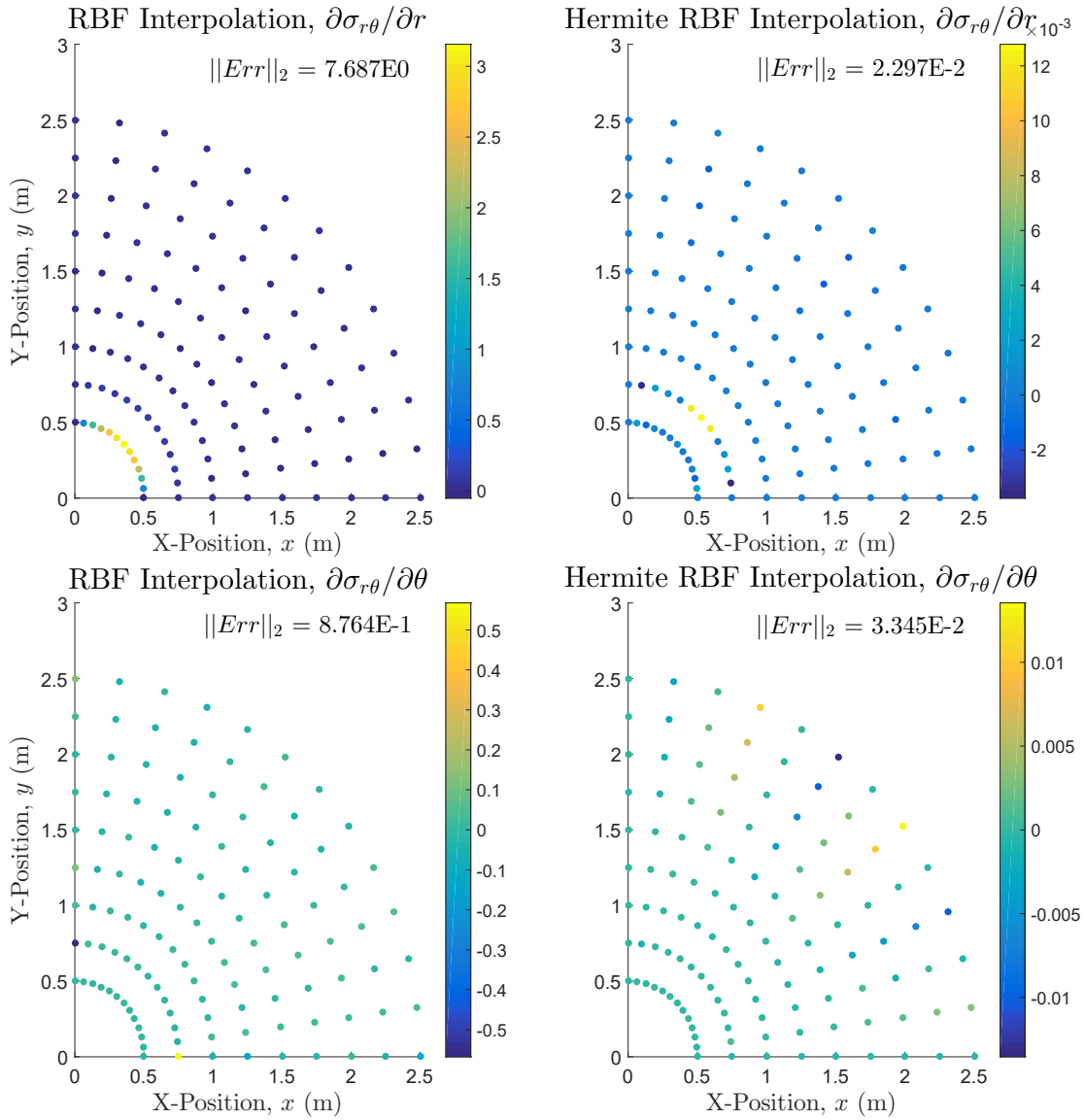


Figure 17: Interpolation centers coloured by the relative error of the radial, circumferential, and shear stress gradients. The Euclidean norm of the error is shown at the top right of each plot.

For the two circumferential gradients ($\partial\sigma_{\theta\theta}/\partial\theta$ and $\partial\sigma_{r\theta}/\partial\theta$), Figure 17 indicates that the Hermite interpolation is an order of magnitude more accurate in the Euclidean norm when compared with the standard RBF technique. Even more telling is the two orders of magnitude of improvement for the two radial gradients

($\partial\sigma_{rr}/\partial r$ and $\partial\sigma_{r\theta}/\partial r$). These trends unequivocally demonstrate the benefits of Hermite interpolation and its extended data field. Of course, including both the stress and stress gradient data entails an increase in computational complexity and storage demands. However, using locally-supported RBFs with small stencils is intended to offset this increase in complexity.

Qualitatively, Figure 17 suggests that Runge's phenomenon is prominent at the domain limits. Specifically, the wave-like circumferential gradient errors appear to be largest at $\theta = 0$ and $\theta = \pi/2$ and decrease in amplitude towards $\theta = \pi/4$. Likewise, the radial gradient errors are largest near the hole at $r = a$ and then decrease in magnitude towards the outer ring of RBF centers at $r = 5a$. Although they are not considered in this thesis, Chebyshev nodes [17] offer a potential remedy of Runge's phenomenon at $\theta = 0,\ \theta = \pi/2,\ r = a$ and $r = 5a$.

5.2.2 Singularity Interpolation

The singular points of a stress tensor field are defined as the locations where each stress component is zero, or $(\sigma_{rr}, \sigma_{\theta\theta}, \sigma_{r\theta}) = (0, 0, 0)$ Pa. For a semi-infinite plate with a circular hole, the Kirsch equations turn up one singular point in the upper-right quadrant at $(r, \theta) = (a, \pi/6)$. Smolik and Skala [41] showed that these singular points include important information about the tensor field, and prior knowledge of their locations may be exploited for more accurate interpolation. To this end, a relatively coarse grid of data centers, with three data centers clustered around the singularity, will be used to extrapolate the stress and stress gradient information to the rest of the domain:

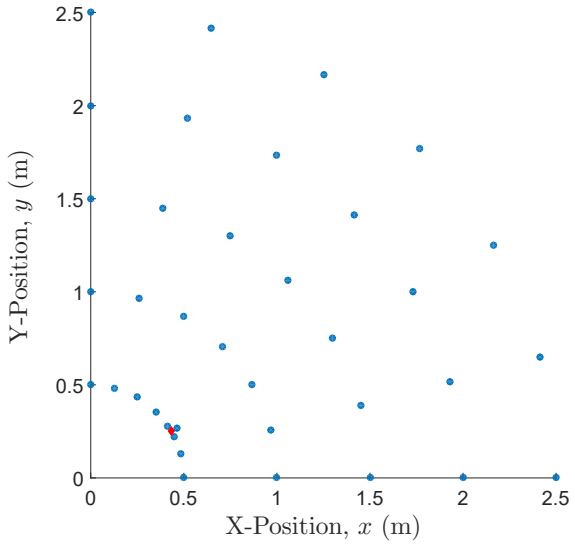
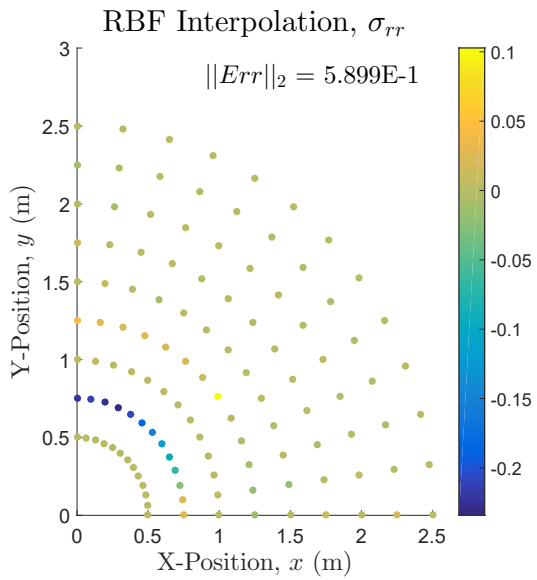


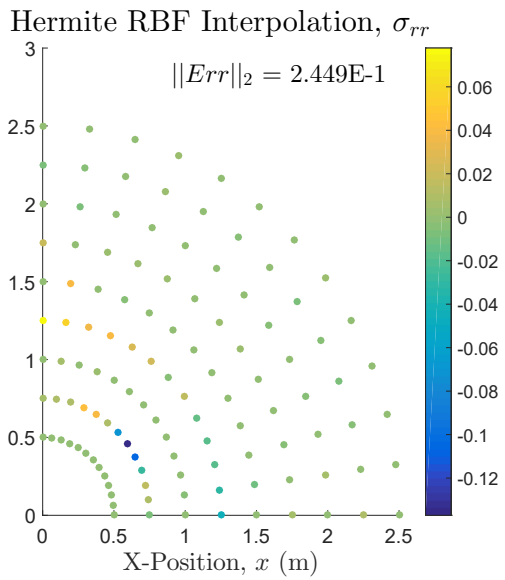
Figure 18: $N = 37$ data centers represented by blue circles and one data center at the singularity represented by a red diamond.

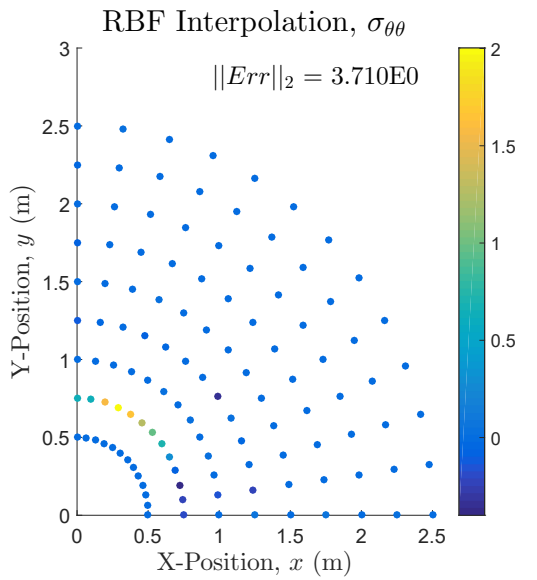
Inputs to the standard and Hermite RBF interpolation algorithms for this test case include:

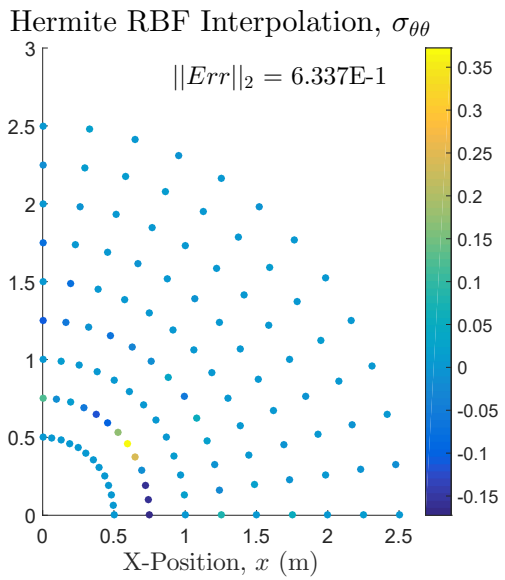
- Shape factor (experimentally determined), $\epsilon = 6$
- Polynomial degree, $d = 3$
- Stencil size, $k = 15$
- Singularity clustering parameter, $R = 0.035$ m

R controls how closely the three RBF centers are clustered around the singularity. This set of inputs results in the following relative errors for the stress fields:


RBF Interpolation, σ_{rr}
$||Err||_2 = 5.899\text{E-}1$
Y-Position, y (m)
X-Position, x (m)


Hermite RBF Interpolation, σ_{rr}
$||Err||_2 = 2.449\text{E-}1$
X-Position, x (m)


RBF Interpolation, $\sigma_{\theta\theta}$
$||Err||_2 = 3.710\text{E}0$
Y-Position, y (m)
X-Position, x (m)


Hermite RBF Interpolation, $\sigma_{\theta\theta}$
$||Err||_2 = 6.337\text{E-}1$
X-Position, x (m)

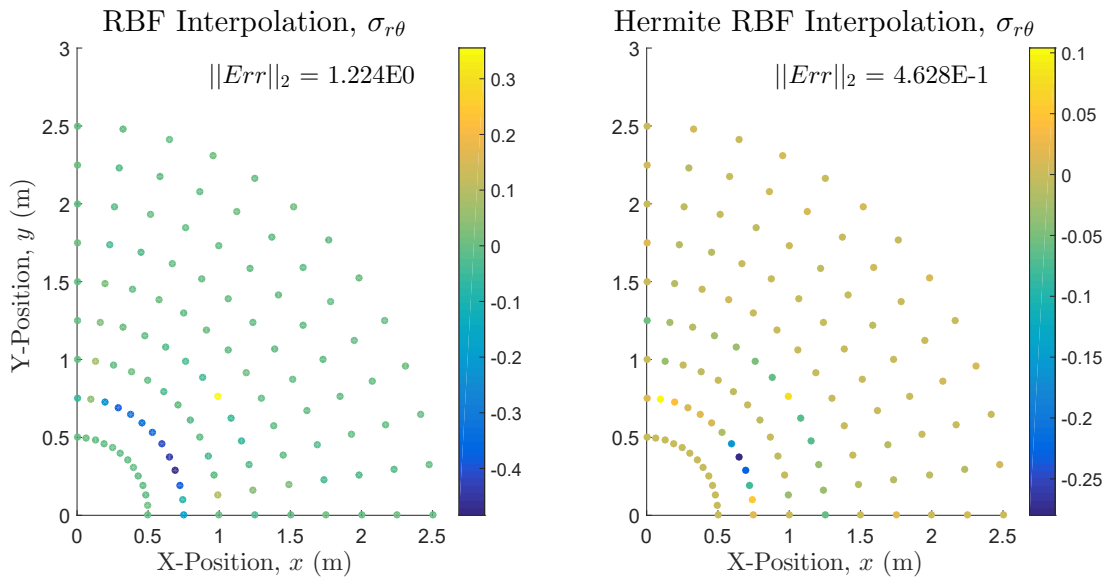
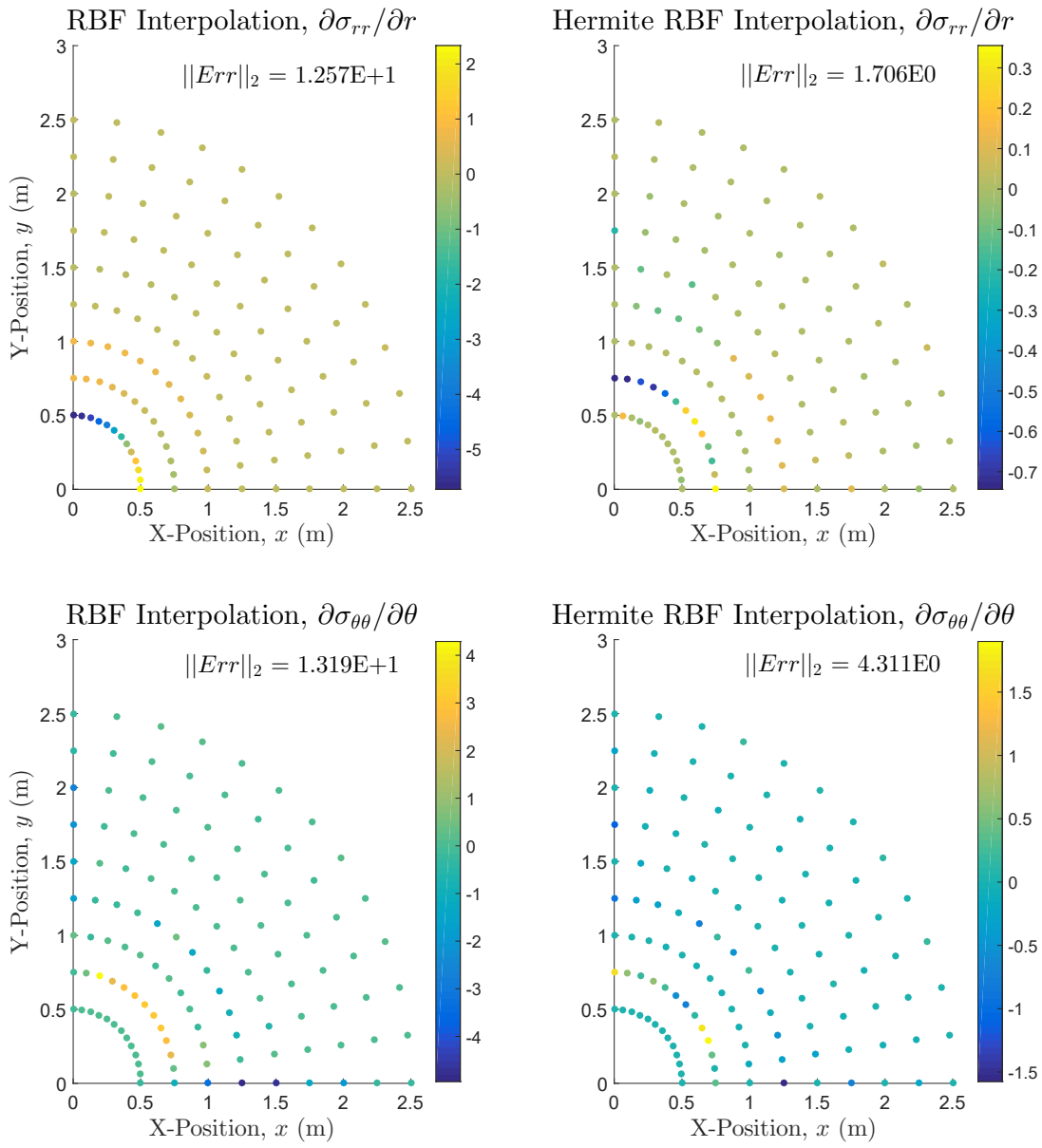


Figure 19: Interpolation centers coloured by the relative error of the radial, circumferential, and shear stress components. The Euclidean norm of the error is shown at the top right of each plot.

Figure 19 suggests that the interpolation errors are collectively much higher than those observed in Figure 16. Intuitively, this trend follows from the comparatively few data centers used to generate Figure 19.

However, it is also evident that opportunistic clustering of the data centers around the singular point gives meaningful results, even for coarse data center layouts. Again, the Hermite interpolation technique outperforms the standard approach by an order of magnitude in accuracy.

The plots beginning on the next page display the relative errors for the stress gradients:


RBF Interpolation, $\partial\sigma_{rr}/\partial r$
$||Err||_2$ = 1.257E+1
Hermite RBF Interpolation, $\partial\sigma_{rr}/\partial r$
$||Err||_2$ = 1.706E0
RBF Interpolation, $\partial\sigma_{\theta\theta}/\partial\theta$
$||Err||_2$ = 1.319E+1
Hermite RBF Interpolation, $\partial\sigma_{\theta\theta}/\partial\theta$
$||Err||_2$ = 4.311E0
Y-Position, y (m)
X-Position, x (m)

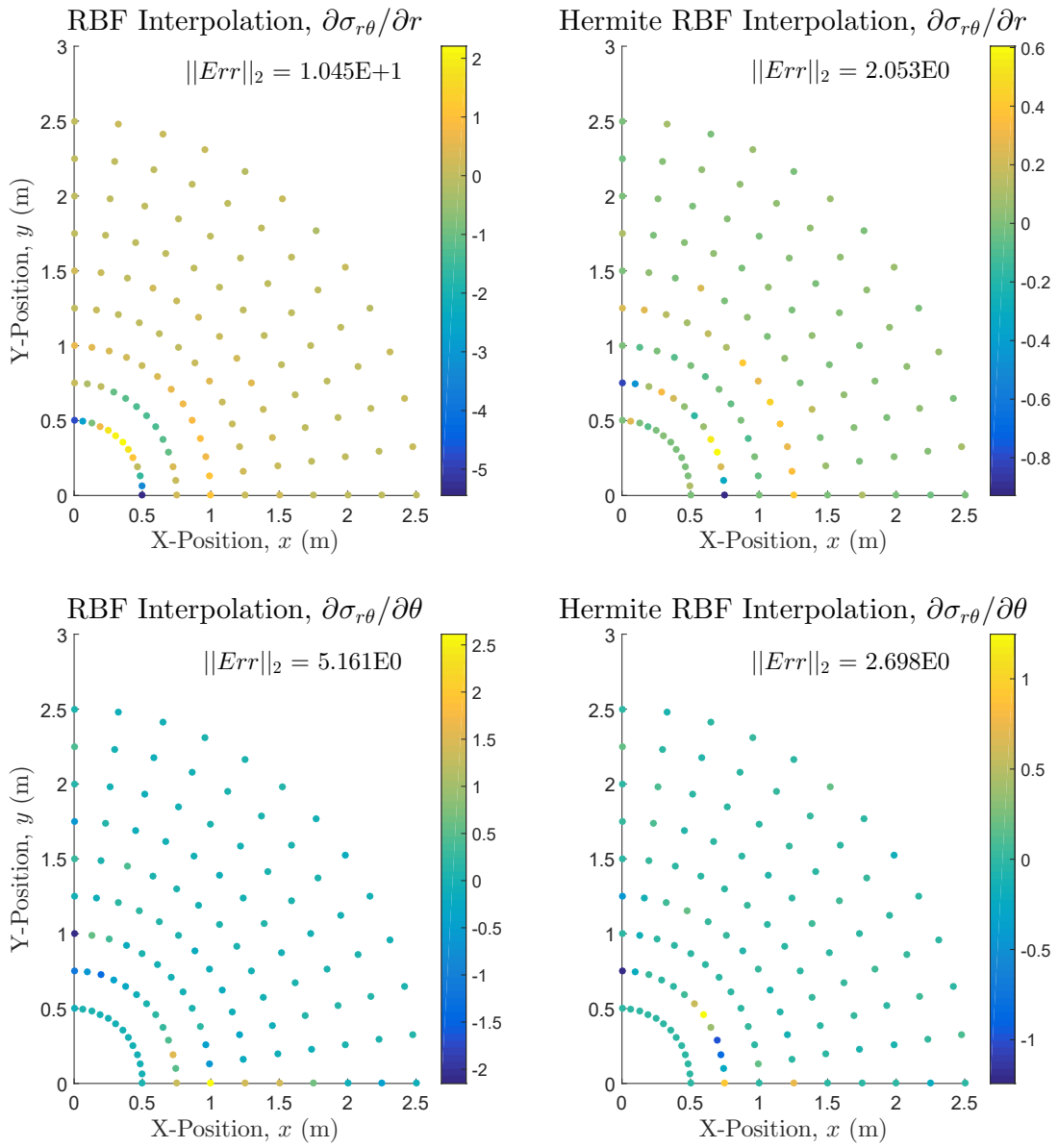


Figure 20: Interpolation centers coloured by the error magnitudes of the radial, circumferential, and shear stress gradients. The Euclidean norm of the error is shown at the top right of each plot.

The stress gradients depicted in Figure 20 are significantly less accurate than those of Figure 17; again an effect of the relatively sparse data centers. For both standard and Hermite RBF interpolation techniques, the largest errors are concentrated near the hole due to Runge's phenomenon. Further, it may be seen that the Hermite RBF interpolation more accurately interpolates the stress gradients in comparison with the standard scheme; a result that is consistent with those seen in Subsection 5.2.1.

Chapter 6

Test Problems: Discretization of PDEs with RBFs

This chapter compares the FDM and RBF discretization techniques by way of three test problems, beginning with the Stefan problem in Section 6.1, followed by the interface sucking problem in 6.2, and concluding with the bubble problem in Section 6.3.

General symbols:

Symbol	**Definition and (Units)**
$\|\| \cdot \|\|_2$	Euclidean norm
$d \in \mathbb{N}$	polynomial degree
$Err(t)$	instantaneous error in the temperature field (K)
$h_{lv} \in \mathbb{R}^+$	heat of vapourization of water (Jkg^{-1})
$k \in \mathbb{N}$	quantity of data centers used in the locally-supported RBFs
$N \in \mathbb{N}$	total quantity of FDM nodes or RBF data centers

$N^p \leq N$	quantity of FDM nodes or RBF data centers lying inside the vapour-phase at the p^{th} time-step
$p \in \mathbb{N}$	time-step index
$P(r,t) \in \mathbb{R}^+$	absolute pressure as a function of radius and time (atm or Pa)
Pe(t) $\in \mathbb{R}^+$	instantaneous Péclet number
$r \in \mathbb{R}^+$	radius measured relative to the center of the bubble (m)
$R(t) \in \mathbb{R}^+$	instantaneous bubble radius (m)
$\dot{R}(t) \in \mathbb{R}$	instantaneous radial bubble velocity (ms^{-1})
$\ddot{R}(t) \in \mathbb{R}$	instantaneous radial bubble acceleration (ms^{-2})
$R_m = 9.912\text{E-}6$	metastable bubble radius (m)
$s(t) \in \mathbb{R}^+$	instantaneous interface position relative to the wall (m)
$t \in \mathbb{R}^+$	time (s)
$u(x,t) \in \mathbb{R}^+$	temperature as a function of position (relative to the wall) and time (K)
$u(\xi,t) \in \mathbb{R}^+$	temperature as a function of position (relative to the interface) and time (K)
$u'(s^{\pm}(t))$ or $\left.\frac{du}{dx}\right\|_{x \to s^{\pm}(t)}$	temperature gradient taken at the left (-) or right (+) side of the interface (Km^{-1})
$\left.\frac{du}{d\xi}\right\|_{\xi \to 0^{\pm}}$	same as above (Km^{-1})
$\frac{Du}{Dt}$	material derivative of the temperature (Ks^{-1})

$v(x,t) \in \mathbb{R}$	velocity as a function of position (relative to the wall) and time ($\mathrm{m}s^{-1}$)
$x \in \mathbb{R}^+$	position relative to the wall (m)
$\mathbf{X}(t^p)$	vector with a length of N^p. Contains the positions of the FDM nodes or RBF data centers in the vapour-phase at the p^{th} time-step (m)

Greek symbols:

Symbol	**Definition and (Units)**
$\alpha \in \mathbb{R}^+$	thermal diffusivity of water ($\mathrm{m^2 s^{-1}}$)
$\beta = \dfrac{\rho_v}{\rho_l}$	ratio of vapour and liquid-phase water densities
$\delta \in \mathbb{R}$	discrepancy between the first and second interface temperature gradient guesses at an arbitrary iteration of the algorithm ($\mathrm{Wm^{-1}}$)
$\Delta t \in \mathbb{R}^+$	uniform time-step size (s)
$\Delta x, \Delta\xi \in \mathbb{R}^+$	uniform length between FDM nodes or RBF data centers (m)
$\epsilon \in \mathbb{R}^+$	RBF shape factor
$\eta =$ 1E-5	initial temperature perturbation (K)
$\kappa \in \mathbb{R}^+$	thermal conductivity of water ($\mathrm{Wm^{-1}K^{-1}}$)
$\mu \in \mathbb{R}^+$	dynamic viscosity of water ($\mathrm{kgm^{-1}s^{-1}}$)
$\rho \in \mathbb{R}^+$	density of water ($\mathrm{kgm^{-3}}$)
$\sigma \in \mathbb{R}^+$	surface tension of liquid water ($\mathrm{Nm^{-1}}$)

$\xi = x - s(t)$	position relative to the interface (m)
$\boldsymbol{\xi}(t^p) \in \mathbb{R}^N$	contains the positions of the FDM nodes or RBF data centers in the liquid-phase at the p^{th} time-step (m)

The subscripts below will be used to indicate a particular state or location. For example $u_v(x,t)$ denotes the temperature in the vapour-phase as a function of position and time:

Symbol	**Definition**
$*$	analytical or semi-analytical value
∞	state infinitely far downstream of the interface
l	liquid-phase
s	phase interface
sat	saturation state
v	vapour-phase
w	wall

6.1 Stefan Problem

The Stefan problem considered here consists of a traveling interface which separates water in its vapour and liquid-phases. Figure 21 is a graphical depiction of the Stefan problem:

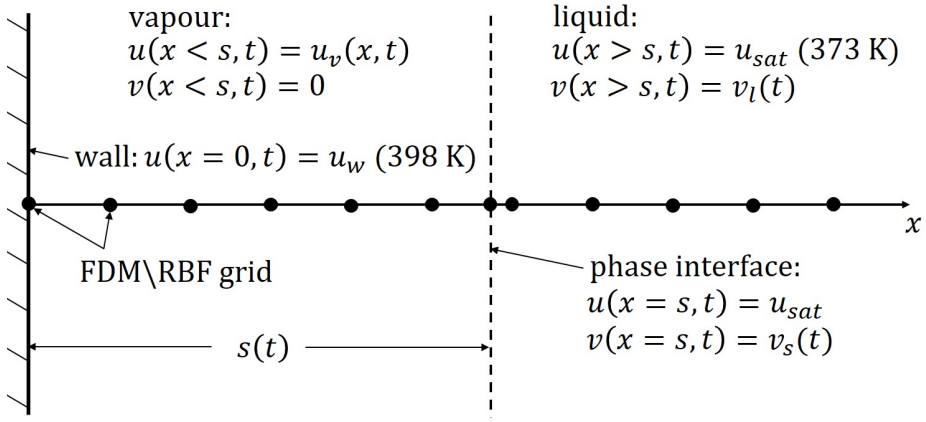


Figure 21: Graphical depiction of the Stefan problem. Given a wall temperature u_w greater than the water saturation temperature u_{sat}, the interface moves in the positive x-direction. Also shown are the FDM and RBF grids to be used for spatial discretization.

Time-evolution of the interface position and velocity are governed by the mass and energy conservation equations. Assuming liquid water to be incompressible, Equation (79) expresses mass conservation in the liquid-phase

$$\frac{\partial v_l}{\partial x} = 0 \tag{79}$$

which implies an instantaneous liquid-phase velocity $v_l(t)$ that is uniform over $x \in [s(t), \infty)$. In conjunction with a mass balance across the phase interface, such that $\rho_l\big(v_l(t) - v_s(t)\big) = -\rho_v v_s(t)$, Equation (79) gives

$$v_l(t) = (1 - \beta)v_s(t) \tag{80}$$

where ρ_l and ρ_v are the liquid and vapour-phase densities of water, $v_s(t)$ is the instantaneous velocity of the interface, and $\beta = \rho_v/\rho_l$. Conservation of energy in the vapour-phase gives

$$\begin{aligned} &\frac{\partial u_v}{\partial t} = \alpha_v \frac{\partial^2 u_v}{\partial x^2},\ x \in \big(0, s(t)\big),\ t \in (0, \infty) \\ &\left.\begin{aligned} &u_v(x = 0, t) = u_w \\ &u_v(x = s(t), t) = u_{sat} \end{aligned}\right\}\ t \in (0, \infty) \\ &u_v(x, t = 0) = u_{sat},\ x \in [0, \infty) \end{aligned} \tag{81}$$

where $u_v(x, t)$ is the temperature in the vapour-phase, α_v is the water vapour thermal diffusivity, $u_w = 398$ K is the wall temperature, $u_{sat} = 373$ K is the water saturation temperature, and $s(t)$ is the instantaneous interface position as measured from the wall. Closure of the heat equation and boundary/initial conditions above is provided by an energy balance across the interface, which relates the interface velocity to the temperature gradients on the vapour and liquid-sides of the interface

$$\rho_v h_{lv} v_s(t) = \kappa_l \left.\frac{du}{dx}\right|_{x \to s^+(t)} - \kappa_v \left.\frac{du}{dx}\right|_{x \to s^-(t)} \tag{82}$$

where κ_l and κ_v are the liquid and vapour-phase thermal conductivities of water and h_{lv} is the heat of vapourization of water. The $x \to s^{\pm}(t)$ notation next to the temperature gradient du/dx indicates that the gradient is taken at the left (-) or right (+) side of the interface. The temperature gradient taken at $s^+(t)$ vanishes because the temperature is spatially uniform in the liquid-phase.

Time discretization of the conservation of energy equation is done using a backward Euler scheme with a time-step size of $\Delta t = 1$ s from $t = 0$ s to $t = 500$ s. Spatial discretization of the Laplacian operator is done using central FDM or RBFs. Namely, RBF discretization follows the four steps on page 70 with $\mathscr{L} \Rightarrow \partial^2/\partial x^2$, and

prescription of the BCs follows Subsection 3.2.1. Both discretization schemes use a stationary grid with nodes and centers[9] located at $x_{i=1,\ldots,N} = (i-1)\Delta x$, with $\Delta x =$ 1E-4 m.

Over the 500 s duration, it was determined experimentally that a total of $N = 400$ nodes/centers is required to track and bound the interface over its full range of motion. At the $(p+1)^{th}$ time-step, only a subset N^{p+1} of those 400 nodes/centers lie inside the vapour-phase domain between $x = 0$ and $x = s(t^{p+1})$. Therefore, the full N x N discrete Laplacian operator may be assembled before the time-marching loop for both FDM and RBF. The appropriate N^{p+1} x N^{p+1} sub-matrix may then be extracted at the $(p+1)^{th}$ time-step for solution of the conservation of energy equation in the vapour-phase. To enforce the Dirichlet boundary condition $u_v\big(s(t^{p+1}), t^{p+1}\big) = u_{sat}$, a node/center must be added at the location of the interface; this step requires minor modifications to the discrete Laplacian operator at each time-step.

Coupling between the interface position and vapour-phase temperature field warrants an iterative solution process. For the p^{th} time-step of the simulation, this solution process is enumerated below:

1. An initial guess, labelled $v_s(t^{p+1})$, is made for the interface velocity at the end of the current time-step.

2. Assuming $v_s(t^{p+1})$ to remain constant from t^p to t^{p+1}, the interface position, as measured from the wall at $x = 0$, is updated by $s(t^{p+1}) = s(t^p) + v_s(t^{p+1})\Delta t$.

3. $v_s(t^{p+1})$ is then used to estimate the temperature gradient at the interface, labelled $\widehat{u}'\big(s^-(t^{p+1})\big)$, via Equation (82).

[9]The terminology "nodes" and "centers" will be used to refer to the FDM and RBF grids, respectively.

4. As mentioned prior, a total of N^{p+1} nodes/centers with locations $(i-1)\Delta x,\ i = 1, ..., N^{p+1}$ lie inside the vapour-phase. A single node/center is placed at $x = s(t^{p+1})$ to enforce the Dirichlet BC and the node/center locations are stored in a vector $\mathbf{X}(t^{p+1})$.

5. The heat equation is solved using the the Dirichlet BCs and one of the two spatial discretization methods described prior. The output of this step is the temperature field in the vapour-phase at the end of the current time-step, $u_v\big(\mathbf{X}(t^{p+1}), t^{p+1}\big)$.

6. $u_v\big(\mathbf{X}(t^{p+1}), t^{p+1}\big)$ is then used to compute a second guess for the interface temperature gradient, called $\overline{u}'\big(s^-(t^{p+1})\big)$. With the FDM scheme this gradient is estimated using a first-order forward difference, whereas the RBF scheme uses the technique outlined on page 72 with $\mathscr{L} \Rightarrow \partial/\partial x$.

7. The discrepancy between the first and second gradient guesses, $\delta = \widehat{u}'\big(s^-(t^{p+1})\big) - \overline{u}'\big(s^-(t^{p+1})\big)$, is measured and $\overline{u}'\big(s^-(t^{p+1})\big)$ is used to update $v_s(t^{p+1})$ via Equation (82).

8. Steps two through seven iterate until the magnitude of δ falls below some preset tolerance. The converged values of $s(t^{p+1}), v_s(t^{p+1}),$ and $u_v\big(\mathbf{X}(t^{p+1}), t^{p+1}\big)$ are stored before moving on to the next time-step.

9. Steps one through eight iterate to march the solution in time.

The plots below display the errors, as functions of time, between the numerical and analytical solutions using both of the spatial discretization techniques. The analytical solutions are denoted with a * superscript and may be found in Appendix F.1. Since the temperature field is a function of both space and time, the temperature error will be succinctly represented by a Euclidean norm at each time t^p

$$||Err(t^p)||_2 = \sum_{i=1}^{N} \sqrt{u(x_i, t^p)^2 - u^*(x_i, t^p)^2} \tag{83}$$

The RBF scheme uses a polynomial with degree $d = 3$, stencil size $k = 9$, and experimentally selected shape factor $\epsilon = 5\text{E}{+}3$ for the Gaussian RBK. The tolerance for δ is set at 1 with a failsafe of 15 maximum iterations in case the algorithm fails to converge. Lastly, the numerical water thermal properties used to generate the solution may be found in Appendix E

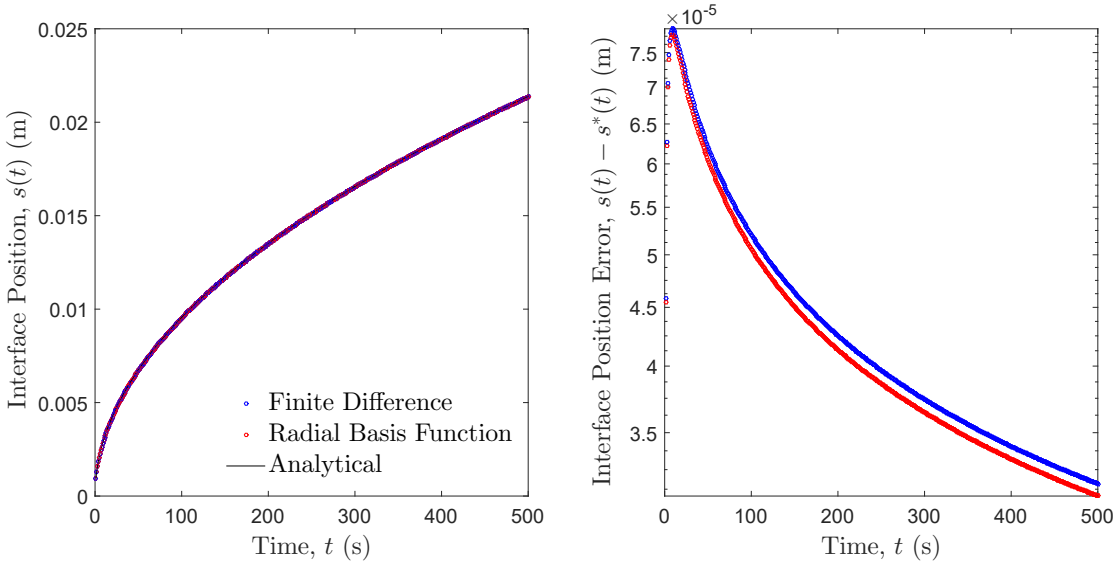


Figure 22: Plots of the interface position (left) and error (right, semi-log) versus time for FDM and RBF spatial discretization methods. The Euclidean norm errors are 1.006E-3 (FDM) and 9.805E-4 (RBF).

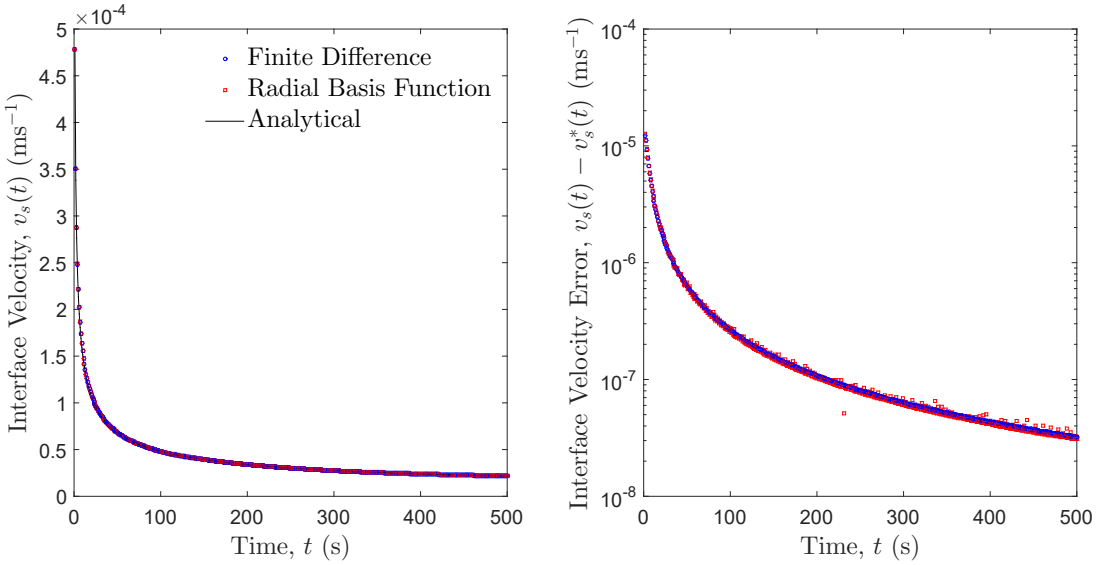


Figure 23: Plots of the interface velocity (left) and error (right, semi-log) versus time for FDM and RBF spatial discretization methods. The Euclidean norm errors are 2.612E-5 (FDM) and 2.650E-5 (RBF).

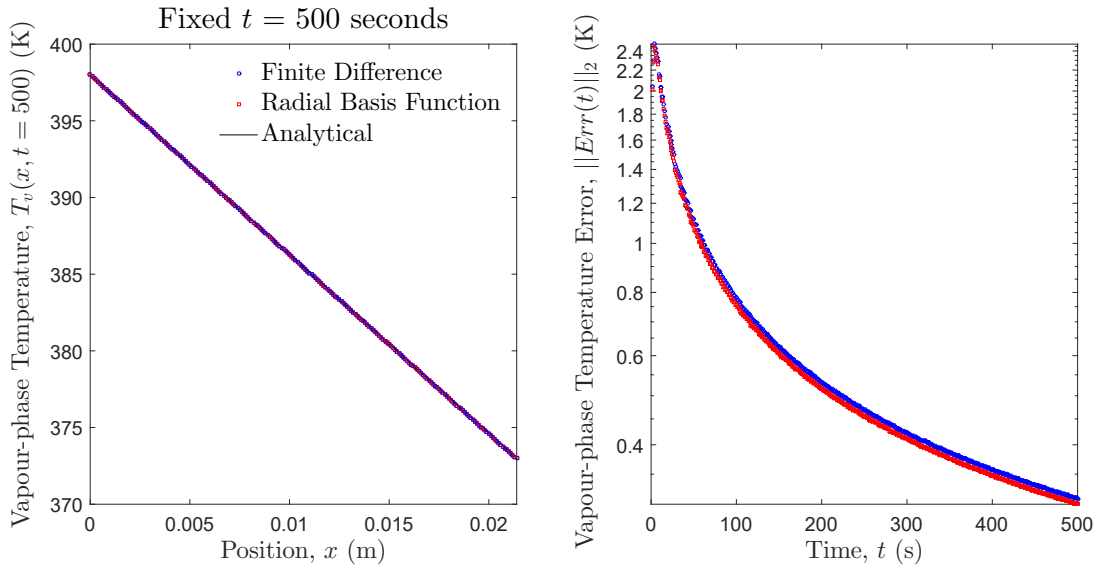


Figure 24: Plots of the vapour-phase temperature field at the final time-step (left) and Euclidean norm error versus time (right, semi-log) for FDM and RBF spatial discretization methods. The error at any given time is calculated using Equation (83).

Qualitatively, Figures 22, 23, and 24 suggest that both the FDM and RBF discretization schemes are able to accurately predict the state of the vapour-liquid system over a 500 second duration. The error plots on the right-hand side of each figure show that,

even with a naively selected shape factor, the RBFs achieve better accuracy for the interface position and temperature field. It is possible that a more mathematically rigorous selection of ϵ would yield even better accuracy.

The superior accuracy of the RBF scheme likely comes from its locally-supported stencil. For an arbitrary node x_i, the FDM stencil with size $k = 3$ includes the node at x_i itself and the two nodes x_{i-1} and x_{i+1} immediately up and downstream. On the other hand, provided that x_i is far removed from either $x = 0$ or $x = s(t)$, the RBF stencil with size $k = 9$ includes the center at x_i and the four centers immediately up and downstream. Sampling the temperature data further up and downstream of x_i involves a more complicated grid topology, which in turn results in a more computationally complex RBF algorithm relative to FDM.

Figure 23 indicates that the interface velocity is more accurately measured using an FDM discretization of the heat equation. Given the intricate coupling between Equations (81) and (82), it is difficult to say why this trend occurs for the velocity but not the position and vapour-phase temperatures.

6.2 Interface Sucking Problem

The interface sucking problem will test the robustness of the RBF discretization scheme by adding a layer of complexity to the Stefan problem. In particular, advection of heat via bulk motion of the liquid-phase must be addressed. By contrast, the Stefan problem considered the temperature field in the motionless vapour-phase. Figure 25 illustrates the fundamentals of the interface sucking problem:

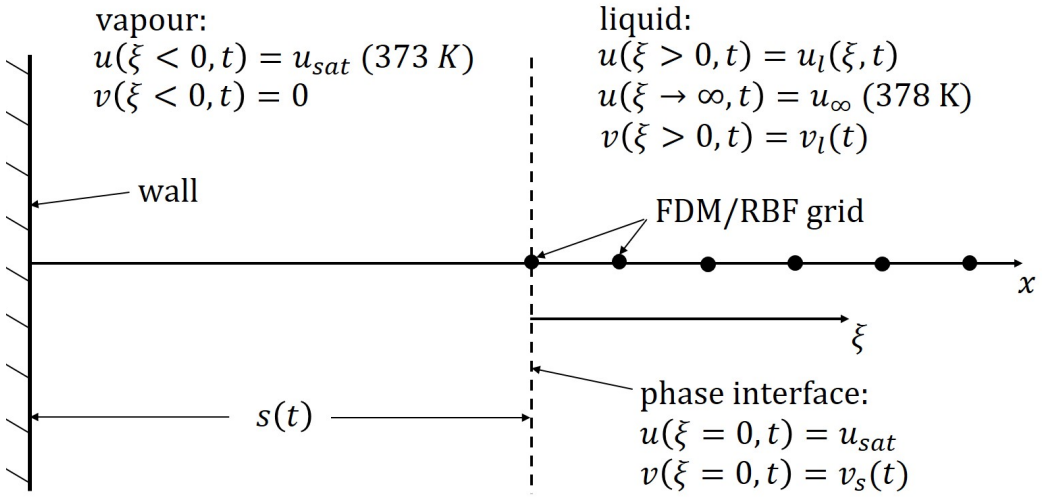


Figure 25: Graphical depiction of the interface sucking problem. A non-inertial reference frame with coordinate ξ moves with the phase interface in the $+\xi$ direction as the liquid water boils. This ξ frame will be adopted for solution of the advection-diffusion equation as in (85).

Enforcing conservation of mass in the incompressible liquid-phase gives the spatially constant velocity $v_l(t) = (1-\beta)v_s(t)$ as measured from the inertial x-reference frame. Application of the energy equation to the liquid-phase may be expressed as follows

$$\begin{aligned} &\frac{\partial u_l}{\partial t} + v_l(t)\frac{\partial u_l}{\partial x} = \alpha_l \frac{\partial^2 u_l}{\partial\, x^2},\ x \in \big(s(t), \infty\big),\ t \in (0, \infty) \\ &\left.\begin{aligned} &u_l(x = s(t), t) = u_{sat} \\ &u_l(x \to \infty, t) = u_\infty \end{aligned}\right\}\ t \in (0, \infty) \\ &u_l(x, t = 0) = u_\infty,\ x \in [0, \infty) \end{aligned} \tag{84}$$

where α_l is the liquid-phase thermal diffusivity, $u_{sat} = 373$ K is the water saturation temperature, and $u_\infty = 378$ K is the temperature infinitely far downstream of the interface.

Discretization of Equation (84) may be simplified by taking the following steps:

- Using a non-inertial reference frame (labeled ξ in Figure 25) with its origin at the interface. ξ is related to the inertial x-reference frame by $\xi = x - s(t)$ and the liquid-phase velocity is accordingly shifted as $v_l'(t) = v_l(t) - v_s(t)$.
- Using a semi-Lagrangian formulation (to be described below).

Implementing the change of coordinates and recasting (84) in Lagrangian form gives

$$\begin{aligned} &\frac{Du_l}{Dt} = \alpha_l \frac{\partial^2 u_l}{\partial \xi^2},\ \xi \in (0,\infty),\ t \in (0,\infty) \\ &\left.\begin{aligned} u_l(\xi = 0, t) &= u_{sat} \\ u_l(\xi \to \infty, t) &= u_\infty \end{aligned}\right\}\ t \in (0,\infty) \\ &u_l(\xi, t = 0) = u_{sat},\ \xi \in [0,\infty) \end{aligned} \tag{85}$$

where D/Dt signifies the material derivative. An energy balance across the interface provides closure of Equation (85) as follows

$$\rho_v h_{lv} v_s(t) = \kappa_l \left.\frac{du}{d\xi}\right|_{\xi \to 0^+} - \kappa_v \left.\frac{du}{d\xi}\right|_{\xi \to 0^-} \tag{86}$$

where the temperature gradient taken at $\xi \to 0^-$ vanishes because the vapour-phase temperature field is spatially uniform.

The solution process for the p^{th} time-step is largely unchanged from that of the Stefan problem; the following sequence of steps replaces steps four and five on page 117:

- A uniformly spaced grid of nodes/centers with locations $(i-1)\Delta\xi,\ i = 1, ..., N$ is used to discretize the liquid-phase. The locations are stored in a vector $\boldsymbol{\xi}(t^{p+1})$.

- The advective part of the energy conservation equation (i.e. the left-hand side of Equation (85)) is discretized with a semi-Lagrangian formulation; see below for details.

- The diffusive part of the energy conservation equation (i.e. the right-hand side of Equation (85)) is discretized with either an FDM or RBF scheme.

- The advective and diffusive parts of the energy conservation equation are combined and solved with the Dirichlet BCs. The output of this step is the liquid-phase temperature field at the end of the current time-step, $u_l\big(\boldsymbol{\xi}(t^{p+1}), t^{p+1}\big)$.

In the semi-Lagrangian scheme, the liquid-phase nodes/centers at time t^{p+1} are advected *backward in time* to estimate their positions at time t^p. From the liquid-phase velocity $v_l(t^{p+1}) = (1-\beta)v_s(t^{p+1})$, the node/center positions at t^p are linearly approximated via $\boldsymbol{\xi}(t^p) = \boldsymbol{\xi}(t^{p+1}) - v_l(t^{p+1})\Delta t$. The liquid-phase temperature field from the previous time-step, which has been stored and labelled as $\widetilde{u}_l\big(\boldsymbol{\xi}(t^p), t^p\big)$, is then interpolated from $\boldsymbol{\xi}(t^p)$ on to $\boldsymbol{\xi}(t^{p+1})$. Labelling this interpolated temperature field as $u_l\big(\boldsymbol{\xi}(t^{p+1}), t^p\big)$, discretization of the advective term may be summarized mathematically like so

$$\left(\frac{Du_l}{Dt}\right)\Delta t \quad \overset{\text{discretize}}{\Longrightarrow} \quad u_l\big(\boldsymbol{\xi}(t^{p+1}), t^{p+1}\big) - \underbrace{\underbrace{\widetilde{u}_l\big(\boldsymbol{\xi}(t^p), t^p\big) + \frac{\partial \widetilde{u}_l}{\partial \xi}}_{\text{Interpolation}} \overbrace{v_l(t^{p+1})\Delta t}^{\text{Advection}}}_{u_l\big(\boldsymbol{\xi}(t^{p+1}), t^p\big)}$$

At this stage, the algebraic expression above may be combined with the discrete diffusion term to evaluate $u_l\big(\boldsymbol{\xi}(t^{p+1}), t^{p+1}\big)$.

A backward Euler scheme is used with a time-step size of $\Delta t = 0.1$ s, starting at $t = 0$ s and ending at $t = 20$ s. A grid of $N = 100$ nodes/centers located at $\xi_{i=1,\ldots,100} = (i-1)\Delta\xi$, with $\Delta\xi = 1\text{E-}4$ m, travels with the phase interface. This

combination of N and $\Delta\xi$ was found to sufficiently capture the thermal boundary layer as the interface moves through its full range of motion.

The remaining input parameters include a Gaussian kernel shape factor, polynomial degree, and stencil size of ϵ = 5E+3, $d = 3$, and $k = 5$ respectively, as well as the numerical water thermal properties in Appendix E. The tolerance for δ is set at 1E-5 with a failsafe of 15 maximum iterations in case the algorithm fails to converge. As before, the analytical solution is indicated with a * superscript and may be found in Appendix F.2. These algorithm inputs yield the following results, where the Euclidean error norm in the temperature field is measured using Equation (83):

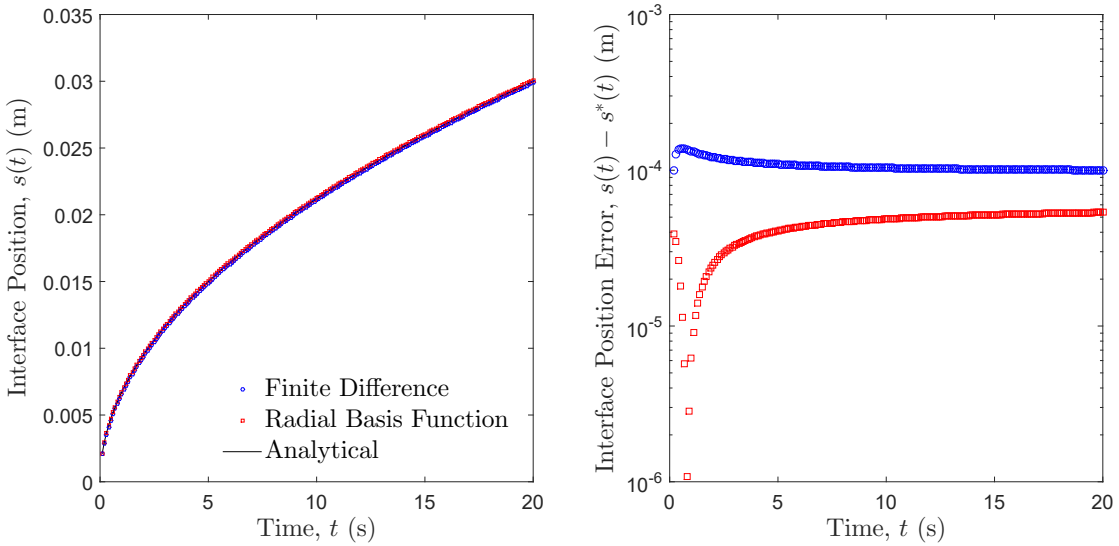


Figure 26: Plots of the interface position (left) and error (right, semi-log) versus time for FDM and RBF spatial discretization methods. The Euclidean norm errors are 2.606E-3 (FDM) and 2.217E-3 (RBF).

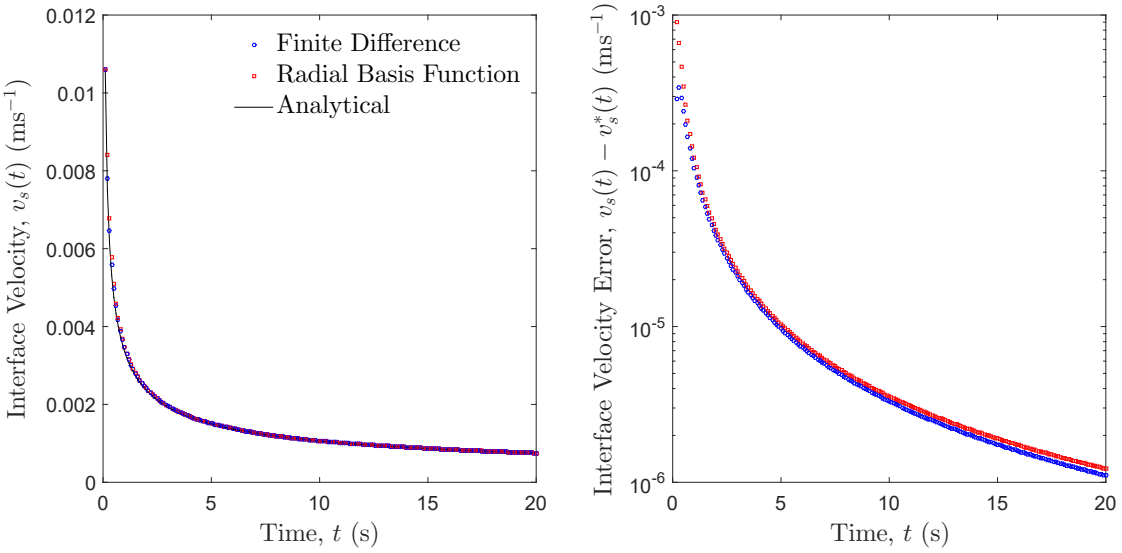


Figure 27: Plots of the interface velocity (left) and error (right, semi-log) versus time for FDM and RBF spatial discretization methods. The Euclidean norm errors are 1.063E-2 (FDM) and 1.069E-2 (RBF).

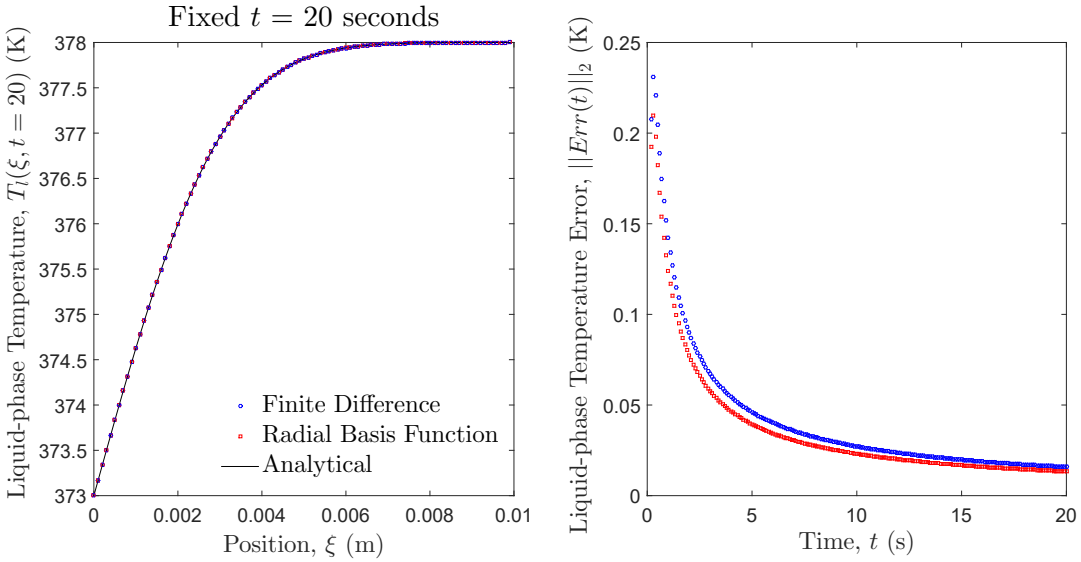


Figure 28: Plots of the liquid-phase temperature field at the final time-step (left) and Euclidean norm error versus time (right) for FDM and RBF spatial discretization methods. The error at any given time is calculated using Equation (83).

Figures 26, 27, and 28 elicit similar trends to the Stefan problem. That is, the RBF discretization scheme appears to resolve the interface position and liquid-phase temperatures more accurately, whereas the interface velocity is more accurately predicted using an FDM scheme.

Care should be exercised when using RBF discretization with Equation (85). In particular, for advection-dominated transport with a high Péclet number, large stencils which sample temperature data far downstream of the interface could cause numerical instability. For reference, the Péclet number for heat transfer is calculated as

$$\mathrm{Pe}(t) = \frac{s(t)v_s(t)}{\alpha_l}$$

and ranges from a maximum of 1.737E+2 at $t = 0.2$ seconds to a minimum of 1.577E+2 at $t = 20$ seconds. Although these numerical instabilities are not explicitly studied here, the figures above imply that such an effect is negligible for the entire range of Péclet numbers. Nevertheless, given a k-stencil centered at ξ_i, a simple remedy would replace the centers furthest downstream of ξ_i with centers upstream of ξ_i. It may also be possible to use shape factors that vary in space or an upwind scheme with biased weighting to upstream centers, but these two possibilities have not been tested.

6.3 Bubble Problem

The model problem discussed in this section introduces another intricacy to the Stefan and interface sucking problems. In particular, the pressure inside a spherical bubble must be computed at each time-step, which necessitates solution of the conservation of momentum equation. The following assumptions are incorporated into the analysis for simplification of the problem [42][43]:

1. The bubble remains spherical as it expands.

2. Pressure and temperature inside the bubble are uniform in space, but both evolve with time.

3. The water vapour contained within the bubble is saturated at all times.

4. Conjointly with assumptions two and three, the temperature field is assumed to be of smoothness class C^0 over the domain $0 \leq r < \infty$. This implies that the temperatures on either side of the interface are approximately equal, or $u(r \to R^-, t) \approx u(r \to R^+, t)$.

5. The liquid water is incompressible.

6. Velocity and temperature in the liquid-phase vary only in the radial dimension (i.e. polar and azimuthal symmetry).

Figure 29 outlines the key concepts of the bubble problem:

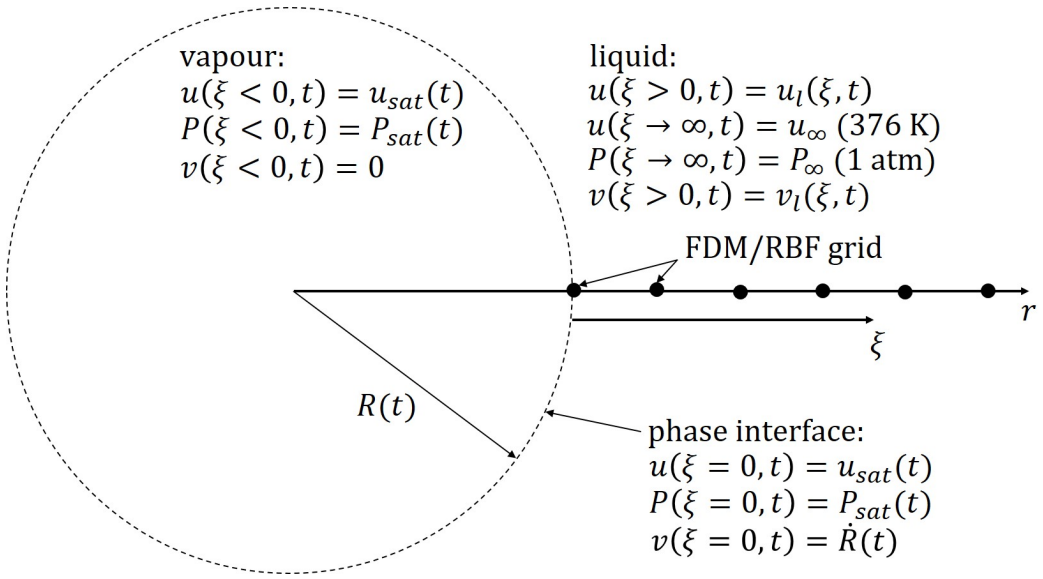


Figure 29: Graphical depiction of the bubble problem. As was done in Section 6.2, a non-inertial frame with coordinate ξ is attached to the phase interface and moves in the $+\xi$ direction as time evolves. This reference frame will be used for discretization of the conservation of energy equation.

Expansion of the bubble is governed by the coupled mass, momentum, and energy conservation equations. In this way, application of the incompressible continuity equation to the liquid-phase may be expressed as follows

$$\frac{\partial}{\partial r}\big(r^2 v_l(r,t)\big) = 0 \tag{87}$$

Combination of Equation (87) with a mass balance across the interface gives the liquid-phase velocity field

$$v_l(r,t) = (1-\beta)\frac{R(t)^2}{r^2}\dot{R}(t) \tag{88}$$

where $R(t)$ and $\dot{R}(t)$ indicate the instantaneous bubble radius and velocity. In similar fashion to Equation (87), conservation of momentum is applied in the liquid-phase with the following boundary and initial conditions

$$\begin{aligned} &\frac{\partial v_l}{\partial t} + v_l(r,t)\frac{\partial v_l}{\partial r} = -\frac{1}{\rho_l}\frac{\partial P_l}{\partial r} + \frac{\mu_l}{\rho_l}\left[\frac{1}{r^2}\left(r^2\frac{\partial v_l}{\partial r}\right) + \frac{2\mu_l}{r^2}\right],\ r \in \big(R(t),\infty\big),\ t \in (0,\infty) \\ &P_l(r\to\infty,t) = P_\infty \end{aligned} \tag{89}$$

where $P_l(r,t)$ is the pressure field in the liquid-phase, μ_l is the dynamic viscosity of liquid water, and $P_\infty = 1$ atm is the pressure infinitely far downstream of the interface. A force balance on the phase interface is necessary to relate the surface and body forces in the liquid-phase to the saturation pressure inside the bubble

$$P_l\big(r = R(t), t\big) = P_{sat}(t) - 4\mu_l(1-\beta)\frac{\dot{R(t)}}{R(t)} - \frac{2\sigma}{R(t)} \tag{90}$$

where σ is the surface tension of the liquid/vapour water interface and $P_{sat}(t)$ is the instantaneous saturation pressure. In words, Equation (90) states that the saturation pressure inside the bubble is balanced by the following forces: the water pressure outside the bubble, the shear force on the bubble arising from the viscosity of the liquid water, and the surface tension which resists elongation of the interface. Consolidation of Equations (89) and (90) returns the Rayleigh-Plesset equation

$$\begin{aligned}
&\rho_l \ddot{R}(t) = \frac{P_{sat}(t) - P_\infty}{R(t)} - \frac{3}{2}\frac{\dot{R}(t)^2}{R(t)} - 4\mu_l(1-\beta)\frac{\dot{R}(t)}{R(t)^2} - \frac{2\sigma}{R(t)^2} \\
&R(t=0) = R_m,\ \dot{R}(t=0) = 0
\end{aligned} \tag{91}$$

where R_m is the metastable bubble radius which satisfies Equation (91) for a motionless bubble with $\dot{R}(0) = \ddot{R}(0) = 0$. In this way, the metastable radius simplifies to $R_m = 2\sigma/(P_{sat}(0) - P_\infty)$. This unintuitive choice for an initial bubble radius is made to avoid singularity of Equation (91) when $R(0) \to 0$.

Conservation of energy is applied to the liquid-phase and returns the advection-diffusion equation, shown below in Lagrangian form. As was done in Section 6.2, a non-inertial reference frame $\xi = r - R(t)$ is attached to the phase interface

$$\begin{aligned}
&\frac{Du_l}{Dt} = \left(\frac{\alpha_l}{R(t)+\xi}\right)\frac{\partial}{\partial \xi}\left[\big(R(t)+\xi\big)\frac{\partial u_l}{\partial \xi}\right],\ \xi \in (0,\infty),\ t \in (0,\infty) \\
&u_l(\xi \to \infty, t) = u_\infty,\ t \in (0,\infty) \\
&u_l(\xi, t=0) = u_\infty,\ \xi \in [0,\infty)
\end{aligned} \tag{92}$$

where $u_\infty = 376$ K is the temperature infinitely far downstream of the bubble. The second boundary condition for Equation (92) arises from an energy balance across the interface

$$h_{lv}\frac{\partial}{\partial t}\left(\frac{4}{3}\pi\rho_l R(t)^3\right) = 4\pi R(t)^2\left(\kappa_l \frac{\partial u}{\partial \xi}\bigg|_{\xi\to 0^+} - \kappa_v \frac{\partial u}{\partial \xi}\bigg|_{\xi\to 0^-}\right) \tag{93}$$

where the vapour-phase temperature gradient at $\xi \to 0^-$ vanishes due to the spatially uniform temperature inside the bubble. Simplification of Equation (92) gives the following

$$\frac{\partial u}{\partial \xi}\bigg|_{\xi\to 0^+} = \frac{\rho_l h_{lv}}{\kappa_l}\dot{R}(t) \tag{94}$$

Closure of Equations (88), (91), and (92) is provided by a state law for saturated water; see Appendix E for details.

The numerical solution procedure is enumerated below, starting at time $t^0 = 0$ s:

1. Since the bubble is initially in a state of equilibrium with $R(0) = R_m$ and $\dot{R}(0) = 0$, a small perturbation of η = 1E-5 K is applied to the vapour temperature. This perturbation effectively creates a force imbalance between the inner and outer surfaces of the interface and results in a small acceleration.

2. Equation (91) is discretized in time using the Newmark-Beta method [44] and then solved iteratively (with the Newton-Raphson method) for the interface position $R(t^1)$ and velocity $\dot{R}(t^1)$. The saturation pressure inside the bubble is temporarily assumed to remain constant over the time-step, such that $P_{sat}(0) = P_{sat}(t^1)$.

3. $\dot{R}(t^1)$ is substituted into Equation (94), which gives the vapour-phase temperature gradient at $\xi \rightarrow 0^+$. This temperature gradient, which constitutes a Neumann BC, is used in conjunction with the Dirichlet BC $u_l(\xi \rightarrow \infty, t) = u_\infty$ for closure of Equation (92).

4. Equation (92) is discretized using backward Euler for the unsteady term, FDM or RBFs for the diffusive term, and a semi-Lagrangian formulation for the advective term. The output of this step is the temperature field in the liquid-phase at the end of the first time-step, $u_l(\boldsymbol{\xi}(t^1), t^1)$.

5. As per assumption four on page 128, the temperature at the bubble interface is taken to be the saturation temperature, or $u(\xi = 0, t^1) = u_{sat}(t^1)$. From $u_{sat}(t^1)$, the vapour pressure $P_{sat}(t^1)$ is computed using the state law for saturated water.

6. The discrepancy $\delta = P_{sat}(t^1) - P_{sat}(0)$, which was assumed to be zero in step two, is measured and steps two through five iterate until δ becomes smaller than a user-specified tolerance. The converged bubble position and velocity are stored for later comparison with the semi-analytical solution.

7. Steps two through six iterate to march the solution in time. The temperature perturbation η is necessary only for the first time-step, with the inertia of the bubble sustaining motion for all subsequent time-steps.

Accurate resolution of the early stages of bubble growth necessitates a fine time-step size. This computationally demanding requirement may be relaxed in the latter stages of growth as the interface velocity approaches zero. In this way, a piecewise set of discretization parameters is specified as follows

$$(\Delta t, N) = \begin{cases} (\text{1E-6 s}, 500), \ t \in [\text{0,1E-3}] \text{ s} \\ (\text{1E-5 s}, 750), \ t \in (\text{1E-3,1E-2}] \text{ s} \end{cases}$$

As was done in Sections 6.1 and 6.2, N is selected such that there are sufficient nodes/centers to fully resolve the thermal boundary layer in the liquid at all times. The grid, which travels with the phase interface, has N nodes/centers located at $\xi_{i=1,\ldots,N} = (i-1)\Delta\xi$, with $\Delta\xi = \text{1E-7 m}$. In addition, a shape factor, polynomial degree, and stencil size of $\epsilon = \text{7E+6}$, $d = 5$, and $k = 11$ are utilized. Lastly, the tolerance for δ is set at 1E-5 with a failsafe of 5 maximum iterations in case the algorithm fails to converge.

Appendix F.3 provides a semi-analytical solution for the interface position and velocity as functions of time. A more rigorous treatment of this solution is left to [43], but it is prudent to point out that the effects of surface tension and viscosity are neglected. The figure below compares the time-evolution of the interface position and velocity for the numerical and semi-analytical solutions:

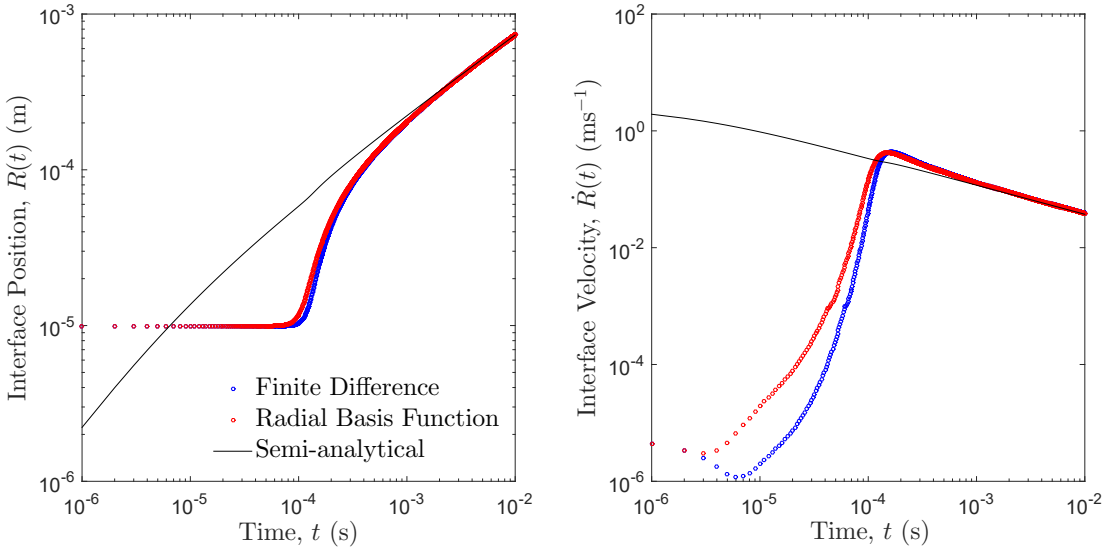


Figure 30: Plots of the interface position (left, log-log) and velocity (right, log-log) versus time for FDM and RBF discretization schemes. Also shown are interface positions and velocities derived from the semi-analytical solution.

Figure 30 prominently shows the effect of omitting the surface tension force in the semi-analytical solution. After the initial period of bubble growth when $R(t)$ is small, the surface tension term in Equation (91) decreases which in turn results in better agreement between the numerical and analytical solutions.

Furthermore, Figure 30 evokes the dynamic behaviour of the bubble in the early stages of the numerical solution. The left-hand plot shows an imperceptibly small change in radius resulting from the 1E-5 K perturbation applied at $t = 0$ seconds, followed by rapid growth. Moreover, it appears that this delay is shorter for the RBF scheme relative to FDM. In the right-hand plot, both RBF and FDM schemes appear to corroborate the latent, but rapid expansion that was noted in the left-hand plot. The RBF interface velocity rapidly catches up with the semi-analytical velocity (which is free from the restraining force of surface tension), and then accurately tracks the velocity in the latter stages of bubble expansion.

A quantitative error analysis would be largely futile given the non-physical behaviour of the semi-analytical solution in the early stages. Rather, the position and

velocity errors are compared below for the final time-step at t = 1E-2 s:

Table 3: Error in the interface position and velocity at the final time-step for FDM and RBF discretization techniques.

	FDM	RBF
Position Error (m)	5.910E-06	4.490E-06
Velocity Error (ms^{-1})	1.477E-03	1.223e-03

which agrees with the previous conclusion that the RBF technique more accurately predicts the interface position and velocity towards the end of the simulation.

Chapter 7

Conclusion

7.1 Research Objectives

The objective of this thesis was to explore the concepts of convex optimization, duality, and radial basis functions and their applications to computational engineering. In Chapter 4, all three of these concepts were combined in the formulation of the LATIN algorithm:

- The error, measured in the energy norm, between the p^{th} flux-temperature guesses and exact (discrete) flux-temperature fields was bounded by the duality gap. As such, the duality gap served as a convergence criterion for the algorithm.
- A one-dimensional, transient thermal diffusion problem was posed as a convex optimization problem; the thermal energy contained in the system was to be minimized and the fluxes and temperatures in the domain were constrained to satisfy the heat equation. Application of the KKT conditions to this problem resulted in SLE (21), which [7], [8], and [15] have shown to be pervasive in science and engineering.

- RBFs were used to discretize the spatial part of the heat equation and the resulting temperature field was compared with FEM discretization.

At each iteration of the LATIN algorithm, a SLE of size n x τ (number of space-steps x number of time-steps) was to be solved for a matrix of temperature corrections. Using ROM and SVD, this SLE was reduced to a size of K x τ with $K \leq \tau$. With parameters of $n = \tau = 10$, K ranging from 1 to 10, and FEM discretization, the LATIN algorithm iteratively converged to a temperature field for each K. These temperature fields were then compared with a semi-analytical temperature field and the Euclidean norm error between them ranged from a maximum of 2.585E0 at $K = 1$ to a minimum of 8.651E-1 at $K = \tau = 10$. This trend was anticipated because $K = 1$ used only the largest singular value, and its pair of left and right singular vectors, to construct the temperature correction field and discarded the other nine. On the other hand, $K = 10$ constructed the temperature correction field using all ten singular values and all ten pairs of left and right singular vectors. However, the larger K were found to be more computationally expensive than the small K, although negligibly so for such a small problem. For industrial-scale problems which may have 10^3 to 10^5 time-steps, it would evidently be best to use a moderate K which balances computational expense and error.

Replacing the FEM with an RBF discretization scheme with all other parameters unchanged, the Euclidean norm errors were measured between the RBF and semi-analytical temperatures at times of 5 and 10 s. Using Gaussian RBFs with stencil size $k = 7$ and polynomial degree $d = 5$, these errors were found to be 9.579E-2 at $t = 5$ s and 7.531E-2 at $t = 10$ s. By contrast, the FEM errors were found to be 1.442E-1 at $t = 5$ s and 1.136E-1 at $t = 10$ s. The superior accuracy of the RBF scheme resulted from the comparatively large k-stencil; the Gaussian RBFs were supported over $k = 7$ nodes whereas the FEM hat functions were supported over $k = 3$ nodes. This

extended support came at a cost, however, as the RBF stiffness matrix was more dense with a bandwidth of seven as opposed to three with FEM.

In Chapter 5, standard and Hermite RBF schemes used given data to interpolate stress and stress gradient tensor fields. In particular, the standard approach used a given set of stress data whereas the Hermite approach used the same stress data *and* a given set of stress gradient data. When the errors between the interpolated and exact fields were compared, it was found that the Hermite scheme outperformed the standard scheme by one order of magnitude for stress and up to two orders for the stress gradients. The Hermite scheme exploited the larger data set and therefore leveraged better accuracy against higher storage costs and algorithm complexity.

Chapter 6 simulated one-dimensional, two-phase heat transfer with three test problems and FDM/RBF discretization of space. The first two of these simulations produced the position and velocity of a vapour/liquid water phase interface as functions of time, as well as the temperature distribution in the vapour-phase (Stefan problem) or liquid-phase (interface sucking problem). The Euclidean norm errors were measured between the numerical and analytical solutions and are summarized in Table 4:

Table 4: Euclidean norm of the errors in interface position and velocity, both taken over all discrete time, and vapour/liquid-phase temperatures for the Stefan and interface sucking problems.

	Stefan Problem		Interface Sucking Problem	
	FDM	RBF	FDM	RBF
Interface Position (m)	1.006E-3	9.805E-4	2.606E-3	2.217E-3
Interface Velocity (ms^{-1})	2.612E-5	2.650E-5	1.063E-2	1.069E-2
Vapour-phase Temperature (K)	1.632E+1	1.595E+1	N/A	N/A
Liquid-phase Temperature (K)	N/A	N/A	8.163E-1	7.153E-1

where the vapour/liquid-phase temperature errors above are taken over all discrete space and time, such that $||Err||_2 = \sum_{i=1}^{N} \sum_{j=1}^{\tau} \sqrt{u(x_i, t_j)^2 - u^*(x_i, t_j)^2}$. As may be seen in the table, the interface position and vapour/liquid-phase temperature distributions were more accurately predicted by the RBF scheme for the Stefan and interface sucking problems. This trend was a result of the generally larger k-stencils used in the RBF scheme relative to the 3-stencils in central FDM. However, as was pointed out repeatedly throughout this thesis, the larger RBF k-stencils involved denser stiffness matrices than FDM and FEM. In spite of its relative complexity, one should keep in mind the meshfree nature and flexibility of data center placement in the RBF approach; two important features which compare favourably with FEM and FDM, respectively. Moreover, intuition would suggest that these two features would be of great value for problems in higher spatial dimensions, although this hypothesis was not tested in this thesis.

Table 4 also shows that FDM more accurately resolved the interface velocity relative to the RBF scheme in the Stefan and interface sucking problems. This result, which was converse to those discussed in the previous paragraph, likely came from the complex coupling between the conservation of mass, momentum, and energy equations. Because of that complexity, a more specific cause for this unintuitive result could not be identified.

The final test of Chapter 6 simulated the expansion of a spherical water bubble in time. Again using FDM and RBFs for spatial discretization of the energy conservation equation, quantitative comparison of these two numerical solutions with the semi-analytical solution was made mostly futile by the latter's negation of surface tension effects. Regardless, the RBF technique was seen to more accurately resolve the position and velocity of the bubble in the latter stages of the simulation. In summary, the results in Chapter 6 demonstrated that RBFs are a viable alternative

to the more popular finite difference, finite element, and finite volume methods in computational heat transfer.

7.2 Recommendations

The research presented in this thesis applied two relatively novel concepts, namely the LATIN algorithm and RBFs, to a series of very simple test problems. These standard toy problems verified the LATIN and RBF concepts as promising alternatives for the numerical solution of engineering problems. The next intuitive step would extend these methods to larger, more complicated geometries in two and three-dimensions. For example, Goldak Technologies Inc.© has applied the LATIN algorithm to the simulation of welds with the VrSuite multiphysics software package.

With respect to the RBFs, a potential next step would blend the interpolation and discretization of respective Chapters 5 and 6 in the simulation of a bubble. An extension of the simulation in Section 6.3, RBF operator discretization would be used to model the two-dimensional, coupled mass, momentum, and energy equations. Taking inspiration from [1], [2], and [3], RBF interpolation would then track the interface as it rises buoyantly and freely deforms from its initial, spherical state.

Section 3.3 explored some of the beneficial properties of the Gaussian RBF and justified its use on those grounds. However, many other RBF types exist, such as the popular multiquadric function and Duchon's thin plate splines. Some or all of the test problems in Chapters 5 and 6 could be solved again using other RBF types and their results compared with those of the Gaussian function.

Perhaps the most significant drawback of the RBF methodology is the shape factor in the infinitely-differentiable class of RBFs. In this thesis, the shape factor was iteratively selected such that the numerical RBF solution agreed "well enough" with

the available analytical solution. However, the majority of problems in computational engineering do not have analytical solutions, making such a rudimentary selection process impossible. It should be noted that a shape factor in the neighbourhood of $\epsilon = O(h^{-1})$, where h is some measure of the distance between data centers, was generally found to yield acceptable accuracy. Thus, a shape factor of that order could be useful as a first approximation in more complex problems. Regardless, a more mathematically rigorous process for selecting ϵ is an active area of research in the RBF community.

One simple workaround to the issue of shape factor selection is to use an RBF from the finitely-differentiable class of functions. Such functions do not have a shape factor but generally have lower error convergence rates compared with the infinitely-differentiable functions. A more promising alternative borrows from the thriving field of machine learning, whereby statistical techniques such as K-fold cross-validation may be used to fine-tune ϵ. A third workaround might use one of the stable basis methods discussed in Section 1.2. As implied by the name, such a basis is stable even in the limit as $\epsilon \to 0$ where the standard Gaussian basis becomes ill-conditioned. However, these bases require decomposition (e.g. HSSVD or QR decomposition) of the collocation matrix and can therefore be computationally expensive.

7.3 Contribution to Knowledge

As was explained in Section 7.1, the research presented in this thesis did not intend to study complex problems with intricate geometries. Rather, the powerful yet under-represented concepts of convex optimization, duality, and RBFs were introduced and tested with a series of toy problems. Moreover, these concepts were tested alongside the more familiar FDM and FEM and compared favourably.

The LATIN algorithm tested in Chapter 4 was inspired by [10], [11], and [12]. This thesis followed a different route and derived the algorithm explicitly from convex optimization and duality. More specifically, LATIN was rebuilt around the discrete KKT conditions in SLE (21) and the duality gap in Equation (49). In [10], [11], and [12], this intuitive and potent interpretation was either left to the imagination of the reader or omitted in favour of addressing other important points.

The growing discipline of mixed FEM is centered around SLE (21) in much the same way as the LATIN algorithm. However, the premise of duality is seldom mentioned in the mixed FEM literature and the value of the duality gap as an error bound is rarely recognized. For this reason, Section 2.5 demonstrated how the duality gap may be cheaply computed by LATIN and then exploited as a criterion for convergence of the algorithm.

Chapter 6 tested the RBF discretization of PDEs, particularly in the simulation of heat transfer between two phases of matter. A series of toy problems demonstrated that the meshfree nature and flexibility of data center placement in the RBF approach were well-suited to modelling advection of heat. By contrast, in Lagrangian and semi-Lagrangian FEM the mesh must be reconstructed at each time-step as the nodes are displaced by bulk fluid motion. As a consequence, large nodal displacements can lead to thin surface elements of poor quality; an issue that is irrelevant in the RBF approach. With regard to FDM, which is best suited to structured, Cartesian grids, bulk fluid motion would tend to disturb that delicate structure in Lagrangian and semi-Lagrangian formulations. For these reasons, RBFs seem to have found some traction in computational fluid dynamics (see [5], [6], [25], [26], and [27]). This thesis attempted, in small part, to extend the RBF discretization technique to the field of computational heat transfer.

List of References

[1] C. Piret, "The orthogonal gradients method: a radial basis functions method for solving partial differential equations on arbitrary surfaces," *Oxford Centre for Collaborative Applied Mathematics*, April 2012.

[2] J. Carr, R. Beatson, J. Cherrie, T. Mitchell, W. Fright, B. McCallum, and T. Evans, "Reconstruction and representation of 3D objects with radial basis functions," *ACM SIGGRAPH*, September 2001.

[3] M. Botsch and L. Kobbelt, "Real-time shape editing using radial basis functions," *EUROGRAPHICS*, vol. 24, 2005.

[4] M. Orr, "Introduction to radial basis function networks," January 1996.

[5] E. Kansa, "Application of Hardy's multiquadric interpolation to hydrodynamics," in *Multiconference on Computer Simulation: Aerospace*, (San Diego, California, United States), January 1986.

[6] E. Kansa, "Multiquadrics - a scattered data approximation scheme with applications to computational fluid-dynamics - II - solutions to parabolic, hyperbolic and elliptic partial differential equations," *Computers & Mathematics with Applications*, vol. 19, pp. 147–161, 1990.

[7] G. Strang, *Introduction to Applied Mathematics*. Wellesley, MA, United States of America: Wellesley-Cambridge Press, 1986.

[8] G. Strang, *Computational Science and Engineering*. Wellesley, MA, United States of America: Wellesley-Cambridge Press, 2007.

[9] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, England: Cambridge University Press, 2004.

[10] W. Prager and J. Synge, "Approximations in elasticity based on the concept of function space," *Quarterly of Applied Mathematics*, vol. 5, pp. 241–269, October 1947.

[11] P. Ladevèze and J. Pelle, *Mastering Calculations in Linear and Nonlinear Mechanics*. Springer Science and Business Media, Inc., 2005.

[12] P. Ladevèze and U. Perego, "Duality preserving discretization of the large time increment methods," *Computational Methods in Applied Mechanics and Engineering*, vol. 189, pp. 205–232, 2000.

[13] G. Maier, "Quadratic programming and theory of elastic-perfectly plastic structures," *Meccanica*, vol. 3, pp. 265–273, December 1968.

[14] H. Touchette, "Legendre-Fenchel transforms in a nutshell." http://appliedmaths.sun.ac.za/~htouchette/articles.html, 2007.

[15] M. Benzi, G. Golub, and J. Liesen, "Numerical solution of saddle point problems," *Acta Numerica*, vol. 14, pp. 1–137, May 2005.

[16] F. Auricchio, F. Brezzi, and C. Lovadina, *Encyclopedia of Computational Mechanics*. John Wiley & Sons, 2004.

[17] B. Fornberg and N. Flyer, *A Primer on Radial Basis Functions with Applications to the Geosciences*. Philadelphia, Pennsylvania, USA: Society for Industrial and Applied Mathematics, 2015.

[18] G. Fasshauer and M. McCourt, *Kernel-Based Approximation Methods using MATLAB*. World Scientific Publishing Co. Pte. Ltd., 19^{th} ed., 2015.

[19] B. Fornberg and C. Piret, "A stable algorithm for flat radial basis functions on a sphere," *SIAM Journal on Scientific Computing*, vol. 30, pp. 60–80, 2007.

[20] B. Fornberg, E. Larsson, and N. Flyer, "Stable computations with Gaussian radial basis functions," *SIAM Journal on Scientific Computing*, vol. 33, pp. 869–892, 2011.

[21] B. Fornberg and C. Piret, "On choosing a radial basis function and a shape parameter when solving a convective PDE on a sphere," *Journal of Computational Physics*, vol. 227, pp. 2758–2780, 2008.

[22] R. Cavoretto, G. Fasshauer, and M. McCourt, "An introduction to the Hilbert-Schmidt SVD using iterated Brownian bridge kernels," *Numerical Algorithms*, vol. 68, pp. 393–422, February 2015.

[23] N. Flyer, B. Fornberg, V. Bayona, and G. Barnett, "On the role of polynomials in RBF-FD approximations: I. interpolation and accuracy," *Journal of Computational Physics*, vol. 321, pp. 21–38, 2016.

[24] N. Flyer, G. Barnett, and L. Wicker, "Enhancing finite differences with radial basis functions: experiments on the Navier-Stokes equations," *Journal of Computational Physics*, vol. 316, pp. 39–62, 2016.

[25] D. Stevens, H. Power, M. Lees, and H. Morvan, "The use of PDE centres in the local RBF Hermitian method for 3D convective-diffusion problems," *Journal of Computational Physics*, vol. 228, pp. 4606–4624, 2009.

[26] D. Stevens, H. Power, and H. Morvan, "An order-N complexity meshless algorithm for transport-type PDEs, based on local Hermitian interpolation," *Engineering Analysis with Boundary Elements*, vol. 33, pp. 425–441, 2009.

[27] D. Stevens, H. Power, M. Lees, and H. Morvan, "A meshless solution technique for the solution of 3D unsaturated zone problems, based on local Hermitian interpolation with radial basis functions," *Transport in Porous Media*, vol. 79, pp. 149–169, 2009.

[28] G. Wright and B. Fornberg, "Scattered node compact finite difference-type formulas generated from radial basis functions," *Journal of Computational Physics*, vol. 212, pp. 99–123, 2006.

[29] Y. Lucet, "The Legendre-Fenchel conjugate: numerical computation." https://people.ok.ubc.ca/ylucet/CCA/man/whatis.htm, 1998.

[30] R. Dunham and K. Pister, "A finite element application of the Hellinger-Reissner variational theorem," in *Proceedings of the Conference on Matrix Methods in Structural Mechanics (2nd)*, (Wright-Patterson Air Force Base, Ohio, United States), October 1968.

[31] J. He, "Derivation of generalized variational principles without using Lagrange multipliers - Part II: Applications to solid mechanics," *Facta Universitatis: Series on Mechanics, Automatic Control and Robotics*, vol. 3, pp. 13–20, 2001.

[32] R. Taylor, "FEAP – a Finite Element Analysis Program, Version 8.5 Theory Manual." http://projects.ce.berkeley.edu/feap/theory85.pdf, 2017.

[33] J. Moreau, "Quadratic programming in mechanics: dynamics of one-sided constraints," *Society of Industrial and Applied Mathematics*, vol. 4, pp. 153–158, 1966.

[34] R. Kalaba and F. Udwadia, "Equations of motion for nonholonomic, constrained dynamical systems via Gauss's principle," *Journal of Applied Mechanics*, vol. 60, pp. 662–668, September 1993.

[35] R. Hardy, "Multiquadric equations of topography and other irregular surfaces," *Journal of Geophysical Research*, vol. 76, pp. 1905–1915, March 1971.

[36] J. Duchon, "Splines minimizing rotation-invariant semi-norms in Sobolev spaces," in *Constructive Theory of Functions of Several Variables*, (Oberwolfach, Germany), April 25 - May 1 1976.

[37] M. Chenoweth, "A local radial basis function method for the numerical solution of partial differential equations," Master of Arts in mathematics, Marshall University, 2012.

[38] W. Madych and S. Nelson, "Bounds on multivariate polynomials and exponential error estimates for multiquadric interpolation," *Journal of Approximation Theory*, vol. 70, pp. 94–114, 1992.

[39] The Mathworks Inc.®, "MATLAB R2015b Documentation." Online software documentation, 2015.

[40] E. Kirsch, "Die theorie der elastizität und die bedürfnisse der festigkeitslehre," *Zeitschrift des Vereines Deutscher Ingenieure*, vol. 42, pp. 797–807, 1898.

[41] M. Smolik and V. Skala, "Vector field interpolation with radial basis functions," in *SIGRAD*, (Visby, Sweden), May 23 - May 24 2016.

[42] T. Kaya, "Analysis of vapor-gas bubbles in a single artery heat pipe," *International Journal of Heat and Mass Transfer*, vol. 52, pp. 5731–5739, September 2009.

[43] H. Lee and H. Merte, "Spherical vapor bubble growth in uniformly superheated liquids," *International Journal of Mass and Heat Transfer*, vol. 39, pp. 2427–2447, 1996.

[44] N. Newmark, "A method of computation for structural dynamics," *Journal of the Engineering Mechanics Division*, vol. 85, pp. 67–94, 1959.

[45] P. Linstrom and W. Mallard, *NIST Chemistry WebBook, NIST Standard Reference Database Number 69*. Gaithersburg MD, United States: National Institute of Standards and Technology, retrieved March 4, 2019.

[46] S. Welch and J. Wilson, "A volume of fluid based method for fluid flows with phase change," *Journal of Computational Physics*, vol. 160, pp. 662–682, 2000.

[47] G. Guédon, "Two-phase heat and mass transfer modeling: flexible numerical methods for energy engineering analysis," doctoral programme in energy and nuclear science and technology, Politecnico Di Milano, 2013.

[48] B. Mikic, W. Rohsenow, and P. Griffith, "On bubble growth rates," *International Journal of Heat and Mass Transfer*, vol. 13, pp. 657–666, April 1970.

Appendix A

Spacetime Discretization of the Hellinger-Reissner Lagrangian Functional

General symbols:

Symbol	Definition and (Units)
$\square_1 * \square_2$	convolution of functions $\square_1$ and $\square_2$
$\langle \square_1, \square_2 \rangle_\square$	inner product of functions $\square_1$ and $\square_2$ over the domain of integration $\square = \Omega$ or Γ
$\mathbf{0}^{mxn}$	matrix of zeros of size m x n
$c_p \in \mathbb{R}^+$	isotropic specific isobaric heat capacity of Ω ($\mathrm{Jkg^{-1}K^{-1}}$)
$d\mathbf{a} = \mathbf{n} \cdot da$	infinitesimal area da with outward-oriented normal vector $\mathbf{n}$ ($\mathrm{m^2}$)
dV	infinitesimal volume ($\mathrm{m^3}$)
grad($\square$)	gradient operator on the field or vector $\square$
$\mathbf{G}$	Gramian matrix

$\mathcal{L}(\mathbf{Q},\mathbf{U}):$ $\mathscr{Q}_h$ x $\mathscr{U}_h \mapsto \mathbb{R}$	Lagrangian function which maps a flux/temperature pair in discrete spacetime to a scalar energy (J)
$m \in \mathbb{N}$	quantity of free primal nodes in Ω and on $\Gamma_{\mathfrak{D}}$
$\widetilde{m} \in \mathbb{N}$	quantity of primal nodes on $\Gamma_{\mathfrak{N}}$ with prescribed Neumann BCs
$M = m + \widetilde{m}$	total quantity of nodes in the primal grid
$n \in \mathbb{N}$	quantity of free dual nodes in Ω and on $\Gamma_{\mathfrak{N}}$
$\widetilde{n} \in \mathbb{N}$	quantity of dual nodes on $\Gamma_{\mathfrak{D}}$ with prescribed Dirichlet BCs
$N = n + \widetilde{n}$	total quantity of nodes in the dual grid
$\mathscr{Q}_h \in \mathbb{R}^{m\text{x}\tau}$	space containing all flux fields which are discrete in spacetime
$\mathbf{q}_{i,k} \in \mathbb{R}$	flux at the i^{th} primal node and k^{th} time-step (Wm^{-2})
$\mathbf{Q} \in \mathscr{Q}_h$	arbitrary flux field in discrete spacetime (Wm^{-2})
$\mathbf{Q}^* \in \mathscr{Q}_h$	exact flux field in discrete spacetime (Wm^{-2})
$\widetilde{\mathbf{Q}} \in \mathbb{R}^{\widetilde{m}\text{x}\tau}$	prescribed Neumann flux field in discrete spacetime (Wm^{-2})
$\mathbf{r} \in \Omega$	position vector (m)
$\mathbf{S} \in \mathbb{R}^{n\text{x}\tau}$	volumetric source/sink field in discrete spacetime (Wm^{-3})
$t \in \mathbb{R}^+$	time (s)
$t_k^{\pm} \in \mathbb{R}^+$	$t_k^+ = \lim_{\epsilon \to 0} t_k + \epsilon,\ t_k^- = \lim_{\epsilon \to 0} t_k - \epsilon$ (s)
$\text{tr}(\square)$	trace of the matrix $\square$

$\mathscr{U}_h \in \mathbb{R}^{n\text{x}\tau}$	space containing all temperature fields which are discrete in spacetime
$u_{i,k} \in \mathbb{R}^+$	temperature at the i^{th} dual node and k^{th} time-step (K)
$\mathbf{U} \in \mathscr{U}_h$	arbitrary temperature field in discrete spacetime (K)
$\mathbf{U}^* \in \mathscr{U}_h$	exact temperature field in discrete spacetime (K)
$\widetilde{\mathbf{U}} \in \mathbb{R}^{\tilde{n}\text{x}\tau}$	prescribed Dirichlet temperature field in discrete spacetime (K)
$\mathbf{U}^\dagger \in \mathscr{U}_h$	temperature lag matrix (K)

Greek symbols:

Symbol	**Definition and (Units)**
$\delta_{k,l}$	Kronecker delta with respect to indices k and l
$\Delta t \in \mathbb{R}^+$	time-step size (s)
$\Gamma = \partial\Omega$	closed boundary of Ω
$\Gamma_D \subseteq \Gamma$	sub-boundary where the Dirichlet BCs are prescribed
$\Gamma_N \subseteq \Gamma$	sub-boundary where the Neumann BCs are prescribed
$\kappa \in \mathbb{R}^+$	isotropic thermal conductivity of Ω ($\text{Wm}^{-1}\text{K}^{-1}$)
$\Omega \subseteq \mathbb{R}^3$	three-dimensional domain
$\phi_i^{\square}(\mathbf{r})$	$\square = q$: basis function for the space component of $\mathscr{Q}_h$ $\square = u$: basis function for the space component of $\mathscr{U}_h$ uses a local coordinate system with its origin at $\mathbf{r}_i$ in space

$\boldsymbol{\phi}^{\square}$	$\square = q$: row vector of length m containing $\phi^{q}_{i=1,\ldots,m}(\mathbf{r})$ $\square = u$: row vector of length n containing $\phi^{u}_{i=\widetilde{n}+1,\ldots,N}(\mathbf{r})$
$\widetilde{\boldsymbol{\phi}}^{\square}$	$\square = q$: row vector of length $\widetilde{m}$ containing $\phi^{q}_{i=m+1,\ldots,M}(\mathbf{r})$ $\square = u$: row vector of length $\widetilde{n}$ containing $\phi^{u}_{i=1,\ldots,\widetilde{n}}(\mathbf{r})$
$\rho \in \mathbb{R}^{+}$	isotropic density of Ω (kgm^{-3})
$\tau \in \mathbb{N}$	quantity of time-steps
$\theta_k(t)$	basis function for the time component of $\mathscr{Q}_h$ and $\mathscr{U}_h$. Uses a local coordinate system with its origin at t_k in time
$\boldsymbol{\theta} \in \mathbb{R}^{\tau+1}$	column vector of length $\tau + 1$ containing $\theta_{k=0,\ldots,\tau}(t)$

Substitution of the vector-matrix products in (17) and (18) into Equation (16) gives the discrete version of the Hellinger-Reissner Lagrangian functional

$$\mathcal{L}(\mathbf{Q},\mathbf{U}) = \int_{t_0}^{t_\tau} \Bigg[\int^{\Omega} \frac{\kappa^{-1}}{2} (\boldsymbol{\phi}^q \mathbf{Q}\boldsymbol{\theta})^{\mathrm{T}} (\boldsymbol{\phi}^q \mathbf{Q}\boldsymbol{\theta}) dV + \int^{\Omega} \mathrm{grad}(\boldsymbol{\phi}^u \mathbf{U}\boldsymbol{\theta})^{\mathrm{T}} (\boldsymbol{\phi}^q \mathbf{Q}\boldsymbol{\theta}) - \int^{\Omega} (\boldsymbol{\phi}^u \mathbf{U}\boldsymbol{\theta})^{\mathrm{T}} \left(\rho c_p \frac{\partial}{\partial t} (\boldsymbol{\phi}^u \mathbf{U}\boldsymbol{\theta}) + \boldsymbol{\phi}^u \mathbf{S}\boldsymbol{\theta} \right) dV - \int^{\Gamma_{\mathfrak{D}}} (\widetilde{\boldsymbol{\phi}}^u \widetilde{\mathbf{U}}\boldsymbol{\theta})^{\mathrm{T}} (\boldsymbol{\phi}^q \mathbf{Q}\boldsymbol{\theta}) \cdot d\mathbf{a} - \int^{\Gamma_{\mathfrak{N}}} (\boldsymbol{\phi}^u \mathbf{U}\boldsymbol{\theta})^{\mathrm{T}} (\widetilde{\boldsymbol{\phi}}^q \widetilde{\mathbf{Q}}\boldsymbol{\theta}) \cdot d\mathbf{a} \Bigg] dt \tag{95}$$

In the interest of a more compact form of (95), the spatial integrations above are evaluated and the resultant expression is rearranged to yield the following

$$\mathcal{L}(\mathbf{Q},\mathbf{U}) = \int_{t_0}^{t_\tau} \boldsymbol{\theta}^{\mathrm{T}} \Bigg[\frac{\kappa^{-1}}{2} \mathbf{Q}^{\mathrm{T}} \left(\int^{\Omega} (\boldsymbol{\phi}^q)^{\mathrm{T}} \boldsymbol{\phi}^q dV \right) \mathbf{Q} + \mathbf{U}^{\mathrm{T}} \left(\int^{\Omega} \mathrm{grad}(\boldsymbol{\phi}^u)^{\mathrm{T}} \boldsymbol{\phi}^q dV \right) \mathbf{Q} - \mathbf{U}^{\mathrm{T}} \left(\int^{\Omega} (\boldsymbol{\phi}^u)^{\mathrm{T}} \boldsymbol{\phi}^u dV \right) \mathbf{S} - \widetilde{\mathbf{U}}^{\mathrm{T}} \left(\int^{\Gamma_{\mathfrak{D}}} (\widetilde{\boldsymbol{\phi}}^u)^{\mathrm{T}} \boldsymbol{\phi}^q \cdot d\mathbf{a} \right) \mathbf{Q} - \mathbf{U}^{\mathrm{T}} \left(\int^{\Gamma_{\mathfrak{N}}} (\boldsymbol{\phi}^u)^{\mathrm{T}} \widetilde{\boldsymbol{\phi}}^q \cdot d\mathbf{a} \right) \widetilde{\mathbf{Q}} \Bigg] \boldsymbol{\theta} dt - \int_{t_0}^{t_\tau} \rho c_p \boldsymbol{\theta}^{\mathrm{T}} \mathbf{U}^{\mathrm{T}} \left(\int^{\Omega} (\boldsymbol{\phi}^u)^{\mathrm{T}} \boldsymbol{\phi}^u dV \right) \mathbf{U} \frac{d\boldsymbol{\theta}}{dt} dt \tag{96}$$

Note that the unsteady term containing $d\boldsymbol{\theta}/dt$ has been isolated above because its forthcoming integration over time will be treated differently than the rest of the right-hand side. The matrix operators in Equation (96), which are indicated with large parentheses ($\square$), inherit a tensor product structure from the $(\boldsymbol{\phi}^{\square})^{\mathrm{T}} \boldsymbol{\phi}^{\square}$. The spatial integrals may be expanded onto these tensor products to give the operators a predictable composition; using the first right-hand side operator for the sake of example

$$\int^{\Omega} (\boldsymbol{\phi}^q)^{\mathrm{T}} \boldsymbol{\phi}^q dV =$$

$$\begin{bmatrix} \int^{\Omega} \phi_1^q(\mathbf{r})\phi_1^q(\mathbf{r})dV & \cdots & \int^{\Omega} \phi_1^q(\mathbf{r})\phi_i^q(\mathbf{r})dV & \cdots & \int^{\Omega} \phi_1^q(\mathbf{r})\phi_m^q(\mathbf{r})dV \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \int^{\Omega} \phi_i^q(\mathbf{r})\phi_1^q(\mathbf{r})dV & \cdots & \int^{\Omega} \phi_i^q(\mathbf{r})\phi_i^q(\mathbf{r})dV & \cdots & \int^{\Omega} \phi_i^q(\mathbf{r})\phi_m^q(\mathbf{r})dV \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \int^{\Omega} \phi_m^q(\mathbf{r})\phi_1^q(\mathbf{r})dV & \cdots & \int^{\Omega} \phi_m^q(\mathbf{r})\phi_i^q(\mathbf{r})dV & \cdots & \int^{\Omega} \phi_m^q(\mathbf{r})\phi_m^q(\mathbf{r})dV \end{bmatrix} \tag{97}$$

By formally defining the inner product as

$$\begin{aligned} \langle f(\mathbf{r}), g(\mathbf{r}) \rangle_{\Omega} &= \int^{\Omega} f(\mathbf{r})g(\mathbf{r})dV, \ \forall \mathbf{r} \in \Omega \\ \langle f(\mathbf{r}), g(\mathbf{r}) \rangle_{\Gamma} &= \int^{\Gamma} f(\mathbf{r})g(\mathbf{r}) \cdot d\mathbf{a}, \ \forall \mathbf{r} \in \Gamma \end{aligned} \tag{98}$$

where $f(\mathbf{r})$ and $g(\mathbf{r})$ are functions defined on Ω and Γ, then the matrix operators in (96) may be classified as Gramian matrices. Henceforth, matrices with a Gramian structure will be denoted by the shorthand $\mathbf{G}(\boldsymbol{\phi}^{\square}, \boldsymbol{\phi}^{\square}, \Omega) = \int^{\Omega} (\boldsymbol{\phi}^{\square})^{\mathrm{T}} \boldsymbol{\phi}^{\square} dV$ or $\mathbf{G}(\boldsymbol{\phi}^{\square}, \boldsymbol{\phi}^{\square}, \Gamma) = \int^{\Gamma} (\boldsymbol{\phi}^{\square})^{\mathrm{T}} \boldsymbol{\phi}^{\square} \cdot d\mathbf{a}$.

Equation (96) may be further simplified by carrying through the integrations over time. For the moment, only the first term in (96) will be considered and the double series notation introduced on page 36 is used

$$\begin{aligned} &\int_{t_0}^{t_\tau} \frac{\kappa^{-1}}{2} \left(\boldsymbol{\theta}^{\mathrm{T}} \mathbf{Q}^{\mathrm{T}} \mathbf{G}(\boldsymbol{\phi}^q, \boldsymbol{\phi}^q, \Omega) \mathbf{Q} \boldsymbol{\theta} \right) dt = \\ &\int_{t_0}^{t_\tau} \frac{\kappa^{-1}}{2} \left[\left(\sum_{i=1}^{m} \sum_{k=0}^{\tau} \mathbf{q}_{i,k} \theta_k(t) \right) \left(\sum_{j=1}^{m} \sum_{l=0}^{\tau} \mathbf{q}_{j,l} \theta_l(t) \right) \langle \phi_i^q(\mathbf{r}), \phi_j^q(\mathbf{r}) \rangle_{\Omega} \right] dt \end{aligned} \tag{99}$$

The right-hand side of the above may be expanded into the following quadruple series

$$\frac{\kappa^{-1}}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}\sum_{k=0}^{\tau}\sum_{l=0}^{\tau}\left(\int_{t_0}^{t_\tau}\theta_k(t)\theta_l(t)dt\right)\mathbf{q}_{i,k}\mathbf{q}_{j,l}\langle\phi_i^q(\mathbf{r}),\phi_j^q(\mathbf{r})\rangle_\Omega \tag{100}$$

At the expense of a minor loss in generality, integration in the temporal domain will hereafter be approximated by a backward Euler scheme. The time-like basis set then takes the form of $\tau+1$ non-overlapping (i.e. orthonormal) rectangular functions. Such a basis $\boldsymbol{\theta}$ is depicted below in both mathematical and graphical (for the case of $\tau=3$) formats

$$\theta_0(t)=\theta(t-t_0)=\begin{cases}1,\ t=t_0\\ 0,\ t\in(t_0^+,t_\tau]\end{cases}$$

$$\theta_k(t)=\theta(t-t_k)=\left.\begin{cases}1,\ t\in(t_{k-1}^+,t_k^-]\\ 0,\ t\in[t_0,t_{k-1}^-]\cup(t_k^+,t_\tau]\end{cases}\right|\ k=1,...,\tau \tag{101}$$

$$\int_{t_0}^{t_\tau}\theta_k(t)\theta_l(t)dt=\delta_{k,l}\Delta t\ \Big|\ k,l=1,...,\tau\quad\text{(orthonormality)}$$

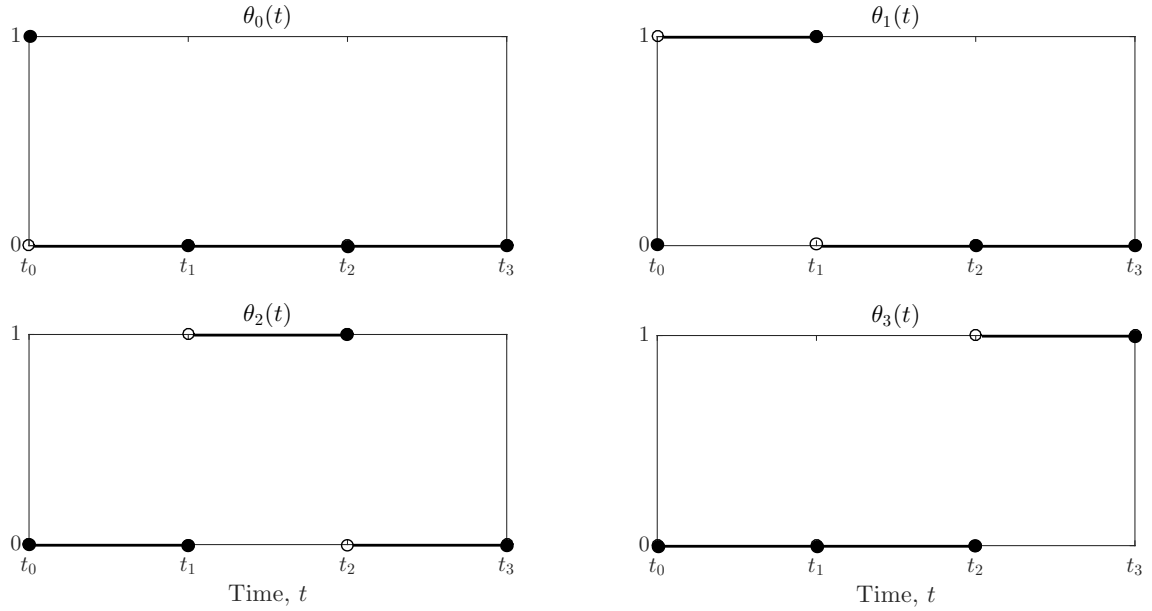


Figure 31: Piecewise constant basis set $\boldsymbol{\theta}=[\theta_0(t)\ \theta_1(t)\ \theta_2(t)\ \theta_3(t)]^{\mathrm{T}}$ corresponding to a backward Euler scheme for the case of $\tau=3$ time-steps each of length Δt.

Note that the $t_\square^-$ and $t_\square^+$ notation used in Equation (101) signifies the limits as t approaches $t_\square$ from the left and right, respectively. Expression (100) reduces to the left-hand side of the equation below via the orthonormality of $\boldsymbol{\theta}$, which then enables the equivalent vector-matrix form below at right

$$\frac{\kappa^{-1}}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}\sum_{k=1}^{\tau}\mathbf{q}_{i,k}\mathbf{q}_{j,k}\langle\phi_i^q(\mathbf{r}),\phi_j^q(\mathbf{r})\rangle_\Omega\Delta t=\frac{\kappa^{-1}}{2}\mathrm{tr}\Big(\mathbf{Q}^{\mathrm{T}}\mathbf{G}(\boldsymbol{\phi}^q,\boldsymbol{\phi}^q,\Omega)\mathbf{Q}\Big)\Delta t \tag{102}$$

where $\mathrm{tr}(\square)$ signifies the trace of a square matrix $\square$. Note also that the summation over $k,l=0,...,\tau$ in expression (100) has been adjusted above to $k=1,...,\tau$. This is because $\theta_0(t)=0$ for all $t\in(t_0,t_\tau]$ as per Equation (101), which consequently maps all $\mathbf{q}_{i,0}$ and $\mathbf{q}_{j,0}$ to zero. On the right-hand side of (102), this adjustment is realized by the deletion of the first column of $\mathbf{Q}$ (and the first row of $\mathbf{Q}^{\mathrm{T}}$); all subsequent iterations of the spacetime fields $\mathbf{Q}$, $\mathbf{U}$, $\widetilde{\mathbf{Q}}$, and $\widetilde{\mathbf{U}}$ will be implicitly understood to include only the last τ columns of the matrices in Equations (17) and (18). The same deletion is implied for the spacetime source/sink field $\mathbf{S}$.

Returning now to Equation (96), special care is required for the integration of the unsteady term containing $d\boldsymbol{\theta}/dt$. As was done previously, summation notation is preferred to the vector/matrix form and admits the following conversion

$$\begin{aligned}&\rho c_p\int_{t_0}^{t_\tau}\left(\boldsymbol{\theta}^{\mathrm{T}}\mathbf{U}^{\mathrm{T}}\mathbf{G}(\boldsymbol{\phi}^u,\boldsymbol{\phi}^u,\Omega)\mathbf{U}\frac{d\boldsymbol{\theta}}{dt}\right)dt=\\&\rho c_p\sum_{\substack{i=\\ \widetilde{n}+1}}^{N}\sum_{\substack{j=\\ \widetilde{n}+1}}^{N}\sum_{k=0}^{\tau}\sum_{l=0}^{\tau}\left(\int_{t_0}^{t_\tau}\theta_k(t)\frac{d\theta_l(t)}{dt}dt\right)u_{i,k}u_{j,l}\langle\phi_i^u(\mathbf{r}),\phi_j^u(\mathbf{r})\rangle_\Omega\end{aligned} \tag{103}$$

It may be observed that the term in parentheses on the right-hand side of Equation (103) represents the *convolution* of θ_k and $d\theta_l/dt$. Moreover, the commutative property of the convolution operation (signified henceforth by $\square*\square$) implies that

$$\theta_k(t)*\frac{d\theta_l(t)}{dt}=\int_{t_0}^{t_\tau}\theta_k(t)\frac{d\theta_l(t)}{dt}dt=\frac{d}{dt}(\theta_k*\theta_l)(t) \tag{104}$$

Working now with the right-most convolution in Equation (104), we wish to first study the behaviour of $(\theta_k * \theta_l)(t)$ for $k, l = 0, ..., \tau$. Recalling that the basis set $\boldsymbol{\theta}$ is comprised of $\tau + 1$ rectangular functions, the convolution is performed by integrating $\theta_k(t)\theta_l(t)$ while keeping θ_k fixed and infinitesimally shifting θ_l over the domain $[t_0, t_\tau]$. This process is repeated for each $k = 0, ..., \tau$, which produces the piecewise linear $(\theta_k * \theta_l)(t)$ depicted below in mathematical and graphical formats

$$
(\theta_{k=0} * \theta_l)(t) = \begin{cases} \Delta t - (t - t_0), \ t \in (t_0^+, t_1^-] \\ 0, \ t = t_0, \ t \in (t_1^+, t_\tau] \end{cases}
$$

$$
(\theta_k * \theta_l)(t) = \left.\begin{cases} 0, \ t \in [t_0, t_{k-1}^-] \\ t - t_{k-1}, \ t \in (t_{k-1}^+, t_k^-] \\ \Delta t - (t - t_k), t \in (t_k^+, t_{k+1}^-] \\ 0, \ t \in (t_{k+1}^+, t_\tau] \end{cases}\right| k = 1, ..., \tau - 1 \qquad (105)
$$

$$
(\theta_{k=\tau} * \theta_l)(t) = \begin{cases} 0, \ t \in [t_0, t_{\tau-1}^-] \\ t - t_{\tau-1}, \ t \in (t_{\tau-1}^+, t_\tau] \end{cases}
$$

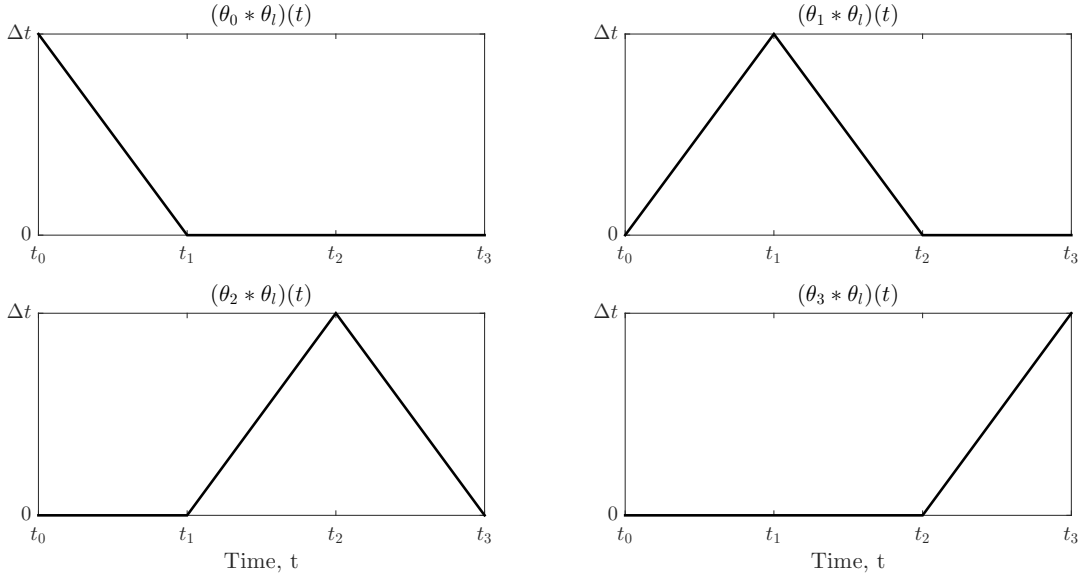


Figure 32: Convolution of the basis set $\boldsymbol{\theta} = [\theta_0(t)\ \theta_1(t)\ \theta_2(t)\ \theta_3(t)]^{\mathrm{T}}$ as a function of time.

The derivative $d/dt(\theta_k * \theta_l)(t)$ follows by inspection of Equation (105) and Figure 33 and is displayed below, again in mathematical and graphical formats

$$\frac{d}{dt}(\theta_{k=0} * \theta_l)(t) = \begin{cases} -1, \ t \in (t_0^+, t_1^-] \\ 0, \ t = t_0, \ t \in (t_1^+, t_\tau] \end{cases}$$

$$\frac{d}{dt}(\theta_k * \theta_l)(t) = \left\{ \begin{array}{l} 0, \ t \in [t_0, t_{k-1}^-] \\ 1, \ t \in (t_{k-1}^+, t_k^-] \\ -1, t \in (t_k^+, t_{k+1}^-] \\ 0, \ t \in (t_{k+1}^+, t_\tau] \end{array} \right| \ k = 1, ..., \tau - 1 \tag{106}$$

$$\frac{d}{dt}(\theta_{k=\tau} * \theta_l)(t) = \begin{cases} 0, \ t \in [t_0, t_{\tau-1}^-] \\ 1, \ t \in (t_{\tau-1}^+, t_\tau] \end{cases}$$

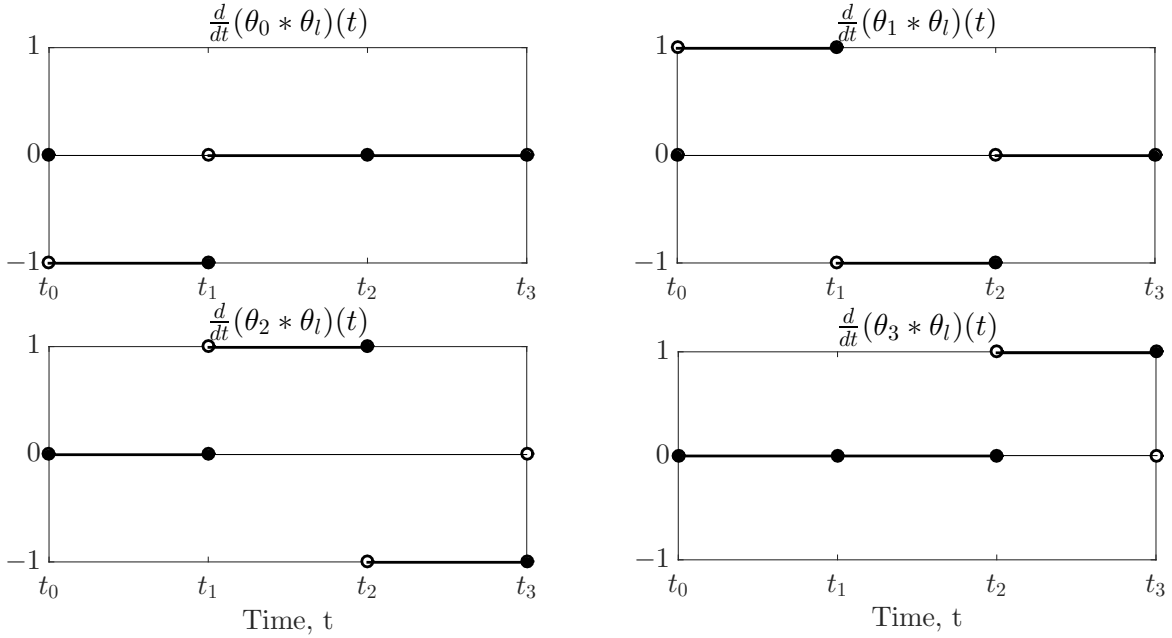


Figure 33: Derivative of the convolution of the basis set $\boldsymbol{\theta} = [\theta_0(t)\ \theta_1(t)\ \theta_2(t)\ \theta_3(t)]^{\mathrm{T}}$ as a function of time.

Substitution of Equation (106) into the right-hand side of Equation (103) gives the expression below

$$\rho c_p \sum_{\substack{i= \\ \widetilde{n}+1}}^{N} \sum_{\substack{j= \\ \widetilde{n}+1}}^{N} \sum_{k=1}^{\tau} u_{i,k}(u_{j,k} - u_{j,k-1}) \langle \phi_i^u(\mathbf{r}), \phi_j^u(\mathbf{r}) \rangle_\Omega \tag{107}$$

which in turn may be converted into vector-matrix form, like so

$$\rho c_p \mathrm{tr} \Big[\mathbf{U}^{\mathrm{T}} \mathbf{G}(\boldsymbol{\phi}^u, \boldsymbol{\phi}^u, \Omega)(\mathbf{U} - \mathbf{U}^\dagger) \Big] \tag{108}$$

where $\mathbf{U}^\dagger$ has the structure shown below[10]

$$\mathbf{U}^\dagger = \begin{bmatrix} u_{\widetilde{n}+1,0} & \cdots & u_{\widetilde{n}+1,k-1} & \cdots & u_{\widetilde{n}+1,\tau-1} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ u_{\widetilde{n}+j,0} & \cdots & u_{\widetilde{n}+j,k-1} & \cdots & u_{\widetilde{n}+j,\tau-1} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ u_{N,0} & \cdots & u_{N,k-1} & \cdots & u_{N,\tau-1} \end{bmatrix} \tag{109}$$

Returning again to Equation (96), time-integration of terms two, three, four, and five on the right-hand side follows the same process as the first term (see Equations (99) - (102)). Substitution of expression (108) for the sixth right-hand side term, combined with the linearity of the trace operation, gives the discrete form of the Hellinger-Reissner Lagrangian function

$$\mathcal{L}(\mathbf{Q}, \mathbf{U}) =$$
$$\mathrm{tr} \Bigg[\frac{\kappa^{-1}}{2} \mathbf{Q}^{\mathrm{T}} \mathbf{G}(\boldsymbol{\phi}^q, \boldsymbol{\phi}^q, \Omega)\mathbf{Q} + \mathbf{U}^{\mathrm{T}} \mathbf{G}(\mathrm{grad}(\boldsymbol{\phi}^u), \boldsymbol{\phi}^q, \Omega)\mathbf{Q} - \frac{\rho c_p}{\Delta t} \mathbf{U}^{\mathrm{T}} \mathbf{G}(\boldsymbol{\phi}^u, \boldsymbol{\phi}^u, \Omega)(\mathbf{U} - \mathbf{U}^\dagger) -$$
$$\mathbf{U}^{\mathrm{T}} \mathbf{G}(\boldsymbol{\phi}^u, \boldsymbol{\phi}^u, \Omega)\mathbf{S} - \widetilde{\mathbf{U}}^{\mathrm{T}} \mathbf{G}(\widetilde{\boldsymbol{\phi}}^u, \boldsymbol{\phi}^q, \Gamma_{\mathfrak{D}})\mathbf{Q} - \mathbf{U}^{\mathrm{T}} \mathbf{G}(\boldsymbol{\phi}^u, \widetilde{\boldsymbol{\phi}}^q, \Gamma_{\mathfrak{N}})\widetilde{\mathbf{Q}} \Bigg] \Delta t \tag{110}$$

[10] Compare $\mathbf{U}^\dagger$ with $\mathbf{U}$ defined in Equation (17).

With the Lagrangian now in discrete form, the KKT and stationarity conditions in (2) and (3) may be applied to (110). In particular, stationarity requires that the partial derivatives of the Lagrangian with respect to the primal and dual variables vanish. From the ensuing set of $(m+n)$ x τ linear equations, the optimal pair of discrete spacetime fields $(\mathbf{Q}^*, \mathbf{U}^*)$ can be extracted. Implementation of $\partial\mathcal{L}/\partial\mathbf{q}_{i,k} = 0,\ i = 1, ..., m,\ k = 1, ..., \tau$ ultimately reduces to the SLE of size m x τ below

$$\kappa^{-1}\mathbf{G}(\boldsymbol{\phi}^q, \boldsymbol{\phi}^q, \Omega)\mathbf{Q} + \mathbf{G}(\text{grad}(\boldsymbol{\phi}^u), \boldsymbol{\phi}^q, \Omega)^{\text{T}}\mathbf{U} - \mathbf{G}(\widetilde{\boldsymbol{\phi}}^u, \boldsymbol{\phi}^q, \Gamma_{\mathfrak{D}})^{\text{T}}\widetilde{\mathbf{U}} = \mathbf{0}^{m\text{x}\tau} \tag{111}$$

Note that the term containing $\widetilde{\mathbf{Q}}$ vanishes under differentiation $\partial\mathcal{L}/\partial\mathbf{q}$ because its entries are constrained by the Neumann BCs. Additionally, the relationships $\mathbf{G}(\text{grad}(\boldsymbol{\phi}^u), \boldsymbol{\phi}^q, \Omega)^{\text{T}} = \mathbf{G}(\boldsymbol{\phi}^q, \text{grad}(\boldsymbol{\phi}^u), \Omega)$ and $\mathbf{G}(\widetilde{\boldsymbol{\phi}}^u, \boldsymbol{\phi}^q, \Gamma_{\mathfrak{D}})^{\text{T}} = \mathbf{G}(\boldsymbol{\phi}^q, \widetilde{\boldsymbol{\phi}}^u, \Gamma_{\mathfrak{D}})$ permit a simpler form of SLE (111)

$$\kappa^{-1}\mathbf{G}(\boldsymbol{\phi}^q, \boldsymbol{\phi}^q, \Omega)\mathbf{Q} + \mathbf{G}(\boldsymbol{\phi}^q, \text{grad}(\boldsymbol{\phi}^u), \Omega)\mathbf{U} = \mathbf{G}(\boldsymbol{\phi}^q, \widetilde{\boldsymbol{\phi}}^u, \Gamma_{\mathfrak{D}})\widetilde{\mathbf{U}} \tag{112}$$

In a similar manner to SLE (111), implementation of $\partial\mathcal{L}/\partial u_{i,k} = 0,\ i = \widetilde{n} + 1, ..., N,\ k = 1, ..., \tau$ ultimately reduces to the SLE of size n x τ below

$$\begin{aligned}&\mathbf{G}(\text{grad}(\boldsymbol{\phi}^u), \boldsymbol{\phi}^q, \Omega)\mathbf{Q} - \frac{\rho c_p}{\Delta t}\mathbf{G}(\boldsymbol{\phi}^u, \boldsymbol{\phi}^u, \Omega)(\mathbf{U} - \mathbf{U}^\dagger) - \mathbf{G}(\boldsymbol{\phi}^u, \boldsymbol{\phi}^u, \Omega)\mathbf{S} - \\ &\mathbf{G}(\boldsymbol{\phi}^u, \widetilde{\boldsymbol{\phi}}^q, \Gamma_{\mathfrak{N}})\widetilde{\mathbf{Q}} = \mathbf{0}^{n\text{x}\tau}\end{aligned} \tag{113}$$

where the term containing $\widetilde{\mathbf{U}}$ vanishes under differentiation $\partial\mathcal{L}/\partial u$ because its entries are constrained by the Dirichlet BCs. To simplify (113), the relationship $\mathbf{G}(\text{grad}(\boldsymbol{\phi}^u), \boldsymbol{\phi}^q, \Omega) = \mathbf{G}(\boldsymbol{\phi}^q, \text{grad}(\boldsymbol{\phi}^u), \Omega)^{\text{T}}$ is exploited to give the following SLE

$$\begin{aligned}&-\mathbf{G}(\boldsymbol{\phi}^q, \text{grad}(\boldsymbol{\phi}^u), \Omega)^{\text{T}}\mathbf{Q} + \frac{\rho c_p}{\Delta t}\mathbf{G}(\boldsymbol{\phi}^u, \boldsymbol{\phi}^u, \Omega)\mathbf{U} = \\ &\frac{\rho c_p}{\Delta t}\mathbf{G}(\boldsymbol{\phi}^u, \boldsymbol{\phi}^u, \Omega)\mathbf{U}^\dagger - \mathbf{G}(\boldsymbol{\phi}^u, \boldsymbol{\phi}^u, \Omega)\mathbf{S} - \mathbf{G}(\boldsymbol{\phi}^u, \widetilde{\boldsymbol{\phi}}^q, \Gamma_{\mathfrak{N}})\widetilde{\mathbf{Q}}\end{aligned} \tag{114}$$

Appendix B

Supplementary Discussion of RBFs and RBKs

General symbols:

Symbol	Definition and (Units)
$\|\| \cdot \|\|_2$	Euclidean norm
$\mathbf{0}^3$	column vector of zeros of length 3
$\mathbf{0}^{3x3}$	matrix of zeros of size 3 x 3
$\mathscr{A} \in \mathbb{R}^2$	given set of scattered data centers, $[\alpha_1 \;\; \alpha_2]^{\mathrm{T}}$
$\mathscr{B} \in \mathbb{R}^2$	given set of scattered x-derivative centers, $[\beta_1 \;\; \beta_2]^{\mathrm{T}}$
$d = 2$	polynomial degree
$\mathscr{K}[x_1] = \mathscr{A}$	the $k = 2$ data centers nearest to x_1
$\mathscr{K}_x[x_1] = \mathscr{B}$	the $k_x = 2$ x-derivative centers nearest to x_1
$k = 2$	quantity of data centers used in $\mathscr{K}[x_1]$

$k_x = 2$	quantity of x-derivative centers used in $\mathscr{K}_x[x_1]$
$m = 1$	dimension of the Euclidean space containing all data and derivative centers
$p_j(x) : \mathbb{R} \mapsto \mathbb{R}$	j^{th} monomial of the quadratic polynomial
$\mathbf{P}(\mathscr{K}[x_1]) \in \mathbb{R}^{2\text{x}3}$	matrix with entries $p_j(x)\Big\|_{x=\alpha_i}$, $i = 1, 2,\ j = 1, 2, 3$
$\mathbf{P}_x(\mathscr{K}[x_1]) \in \mathbb{R}^{2\text{x}3}$	matrix with entries $\dfrac{\partial p_j(x)}{\partial x}\Big\|_{x=\beta_i}$, $i = 1, 2,\ j = 1, 2, 3$
$s(x) \in \mathbb{R} \mapsto \mathbb{R}$	scalar interpolant
$x \in \mathbb{R}$	position
$x_1 \in \mathbb{R}$	interpolation center in 1-dimensional space
$z = 3$	number of monomials in the one-dimensional quadratic polynomial

Greek symbols:

Symbol	**Definition and (Units)**
$\alpha_i \in \mathscr{A}$	data center in 1-dimensional space
$\beta_i \in \mathscr{B}$	x-derivative center in 1-dimensional space
$\epsilon = 1$	RBF shape factor
$\phi(x, \alpha_j) :$ $\mathbb{R}$ x $\mathbb{R} \mapsto \mathbb{R}$	the RBK evaluated at x and centered at α_j

$\mathbf{\Phi}(\mathscr{K}[x_1], \mathscr{K}[x_1]) \in \mathbb{R}^{2\text{x}2}$	matrix with entries $\phi(x, \alpha_j)\Big\|_{x=\alpha_i}$, $i = 1, 2,\ j = 1, 2$
$\dfrac{\partial \phi(x, \beta_j)}{\partial x}$: $\mathbb{R}$ x $\mathbb{R} \mapsto \mathbb{R}$	x-derivative of the RBK evaluated at x and centered at β_j
$\mathbf{\Phi}_x(\mathscr{K}[x_1], \mathscr{K}_x[x_1]) \in \mathbb{R}^{2\text{x}2}$	matrix with entries $\dfrac{\partial \phi(x, \beta_j)}{\partial x}\Big\|_{x=\alpha_i}$, $i = 1, 2,\ j = 1, 2$
$\dfrac{\partial \phi(x, \alpha_j)}{\partial x}$: $\mathbb{R}$ x $\mathbb{R} \mapsto \mathbb{R}$	x-derivative of the RBK evaluated at x and centered at α_j
$\mathbf{\Phi}_x(\mathscr{K}_x[x_1], \mathscr{K}[x_1]) \in \mathbb{R}^{2\text{x}2}$	matrix with entries $\dfrac{\partial \phi(x, \alpha_j)}{\partial x}\Big\|_{x=\beta_i}$, $i = 1, 2,\ j = 1, 2$
$\dfrac{\partial^2 \phi(x, \beta_j)}{\partial x^2}$: $\mathbb{R}$ x $\mathbb{R} \mapsto \mathbb{R}$	xx-derivative of the RBK evaluated at x and centered at β_j
$\mathbf{\Phi}_{xx}(\mathscr{K}_x[x_1], \mathscr{K}_x[x_1]) \in \mathbb{R}^{2\text{x}2}$	matrix with entries $\dfrac{\partial^2 \phi(x, \beta_j)}{\partial x^2}\Big\|_{x=\beta_i}$, $i = 1, 2,\ j = 1, 2$

In the standard RBF interpolation of scattered data, the collocation and vanishing moment constraints result in SLE (55) reprinted below

$$\begin{bmatrix} \mathbf{\Phi}(\mathscr{K}[\mathbf{r}], \mathscr{K}[\mathbf{r}]) & \mathbf{P}(\mathscr{K}[\mathbf{r}]) \\ \mathbf{P}(\mathscr{K}[\mathbf{r}])^{\mathrm{T}} & \mathbf{0}^{zxz} \end{bmatrix} \begin{bmatrix} \mathbf{w} \\ \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} \mathscr{F}[\mathbf{r}] \\ \mathbf{0}^{z} \end{bmatrix}$$

To see the structure of the sub-matrices above, consider an $m = 1$-dimensional test with the following parameters:

- Data center coordinate set: $\mathscr{A} = [\alpha_1 \;\; \alpha_2]^{\mathrm{T}}$.
- Interpolation center (location where $s(\mathbf{r} = x)$ is to be approximated): x_1.
- Stencil size: $k = 2$.
- Gaussian RBK: $\phi(x, \alpha_i) = \exp[-\epsilon^2(||x - \alpha_i||_2)^2]$.
- Shape factor: $\epsilon = 1$.
- Quadratic polynomial basis: $p_j(x) = [p_0(x) \;\; p_1(x) \;\; p_2(x)] = [1 \;\; x \;\; x^2]$.

With these input parameters, the 2-stencil of x_1 consists of its nearest neighbouring data centers, thus $\mathscr{K}[x_1] = [\alpha_1 \;\; \alpha_2]^{\mathrm{T}}$. The $\mathbf{\Phi}$ and $\mathbf{P}$ matrices for this stencil have the following entries

$$\mathbf{\Phi}(\mathscr{K}[x_1], \mathscr{K}[x_1]) = \begin{bmatrix} \phi(x, \alpha_1)\Big|_{x=\alpha_1} & \phi(x, \alpha_2)\Big|_{x=\alpha_1} \\ \phi(x, \alpha_1)\Big|_{x=\alpha_2} & \phi(x, \alpha_2)\Big|_{x=\alpha_2} \end{bmatrix}$$

$$\mathbf{P}(\mathscr{K}[x_1]) = \begin{bmatrix} 1\Big|_{x=\alpha_1} & x\Big|_{x=\alpha_1} & x^2\Big|_{x=\alpha_1} \\ 1\Big|_{x=\alpha_2} & x\Big|_{x=\alpha_2} & x^2\Big|_{x=\alpha_2} \end{bmatrix}$$

The $\phi(x,\alpha)$ convention used for the RBKs is illustrated graphically in Figure 34:

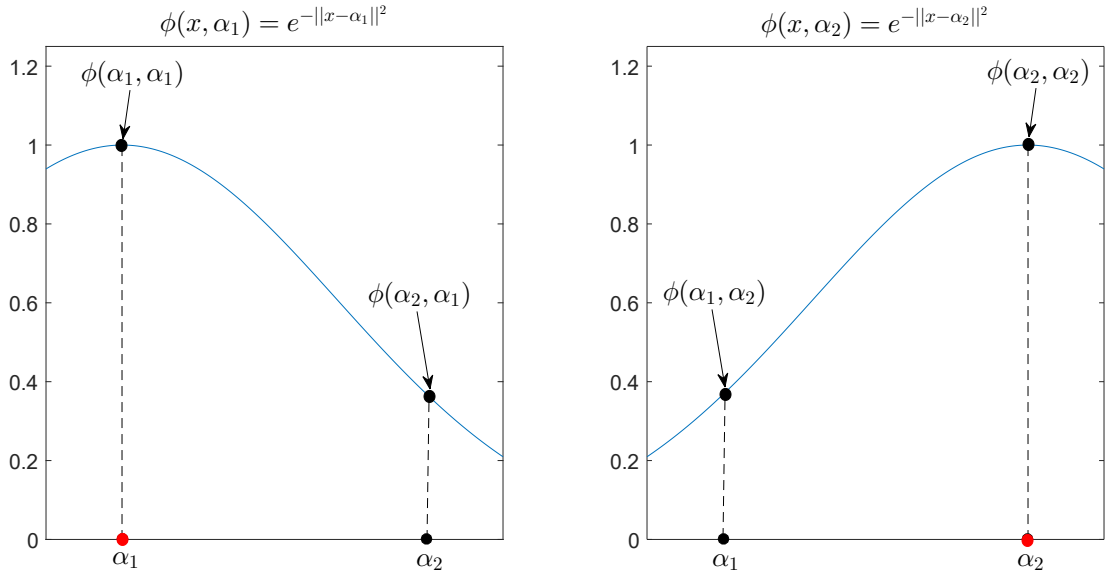


Figure 34: Gaussian functions centered at α_1 (left) and α_2 (right). Also shown are the Gaussian kernels $\phi(\alpha_i,\alpha_j)$ which generate the sub-matrix $\mathbf{\Phi}(\mathscr{K}[x_1],\mathscr{K}[x_1])$.

In the Hermite interpolation scheme, the collocation and vanishing moment constraints result in SLE (58) reprinted below

$$\begin{bmatrix} \mathbf{\Phi}(\mathscr{K}[\mathbf{r}],\mathscr{K}[\mathbf{r}]) & \mathbf{\Phi}_x(\mathscr{K}[\mathbf{r}],\mathscr{K}_x[\mathbf{r}]) & \mathbf{P}(\mathscr{K}[\mathbf{r}]) \\ \mathbf{\Phi}_x(\mathscr{K}_x[\mathbf{r}],\mathscr{K}[\mathbf{r}]) & \mathbf{\Phi}_{xx}(\mathscr{K}_x[\mathbf{r}],\mathscr{K}_x[\mathbf{r}]) & \mathbf{P}_x(\mathscr{K}_x[\mathbf{r}]) \\ \mathbf{P}(\mathscr{K}[\mathbf{r}])^{\mathrm{T}} & \mathbf{P}_x(\mathscr{K}_x[\mathbf{r}])^{\mathrm{T}} & \mathbf{0}^{z\mathrm{x}z} \end{bmatrix} \begin{bmatrix} \mathbf{w} \\ \mathbf{w}_x \\ \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} \mathscr{F}[\mathbf{r}] \\ \mathscr{F}_x[\mathbf{r}] \\ \mathbf{0}^z \end{bmatrix}$$

To generate the $\mathbf{\Phi}_x$, $\mathbf{\Phi}_{xx}$ and $\mathbf{P}_x$ sub-matrices above, a one-dimensional set of derivative centers $\mathscr{B} = [\beta_1 \;\; \beta_2]^{\mathrm{T}}$ is added to the data center set $\mathscr{A}$. In accordance, an additional stencil must be assembled from the derivative centers nearest to x_1. Choosing a stencil size of $k_x = 2$ yields $\mathscr{K}_x[x_1] = [\beta_1 \;\; \beta_2]^{\mathrm{T}}$ and the following sub-matrices

$$\boldsymbol{\Phi}_x(\mathscr{K}[x_1], \mathscr{K}_x[x_1]) = \begin{bmatrix} \left.\dfrac{\partial \phi(x,\beta_1)}{\partial x}\right|_{x=\alpha_1} & \left.\dfrac{\partial \phi(x,\beta_2)}{\partial x}\right|_{x=\alpha_1} \\ \left.\dfrac{\partial \phi(x,\beta_1)}{\partial x}\right|_{x=\alpha_2} & \left.\dfrac{\partial \phi(x,\beta_2)}{\partial x}\right|_{x=\alpha_2} \end{bmatrix}$$

$$\boldsymbol{\Phi}_x(\mathscr{K}_x[x_1], \mathscr{K}[x_1]) = \begin{bmatrix} \left.\dfrac{\partial \phi(x,\alpha_1)}{\partial x}\right|_{x=\beta_1} & \left.\dfrac{\partial \phi(x,\alpha_2)}{\partial x}\right|_{x=\beta_1} \\ \left.\dfrac{\partial \phi(x,\alpha_1)}{\partial x}\right|_{x=\beta_2} & \left.\dfrac{\partial \phi(x,\alpha_2)}{\partial x}\right|_{x=\beta_2} \end{bmatrix}$$

$$\boldsymbol{\Phi}_{xx}(\mathscr{K}_x[x_1], \mathscr{K}_x[x_1]) = \begin{bmatrix} \left.\dfrac{\partial^2 \phi(x,\beta_1)}{\partial x^2}\right|_{x=\beta_1} & \left.\dfrac{\partial^2 \phi(x,\beta_2)}{\partial x^2}\right|_{x=\beta_1} \\ \left.\dfrac{\partial^2 \phi(x,\beta_1)}{\partial x^2}\right|_{x=\beta_2} & \left.\dfrac{\partial^2 \phi(x,\beta_2)}{\partial x^2}\right|_{x=\beta_2} \end{bmatrix}$$

$$\mathbf{P}_x(\mathscr{K}[x_1]) = \begin{bmatrix} \left.\dfrac{\partial}{\partial x}1\right|_{x=\beta_1} & \left.\dfrac{\partial}{\partial x}x\right|_{x=\beta_1} & \left.\dfrac{\partial}{\partial x}x^2\right|_{x=\beta_1} \\ \left.\dfrac{\partial}{\partial x}1\right|_{x=\beta_2} & \left.\dfrac{\partial}{\partial x}x\right|_{x=\beta_2} & \left.\dfrac{\partial}{\partial x}x^2\right|_{x=\beta_2} \end{bmatrix}$$

As was done for the standard RBF scheme, Figures 35, 36, and 37 graphically depict the RBK derivatives above:

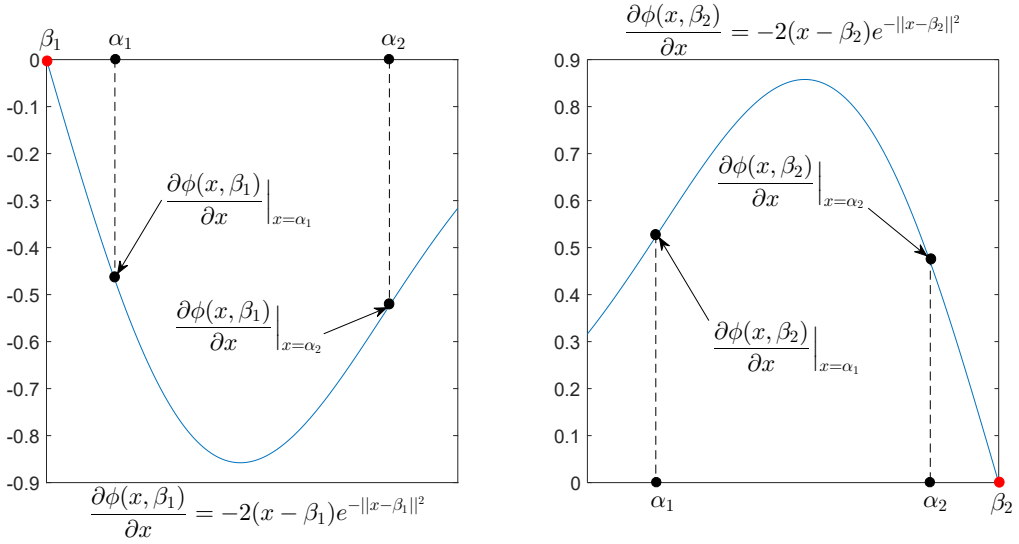


Figure 35: First-order derivatives of the Gaussian functions centered at β_1 (left) and β_2 (right). Also shown are the Gaussian kernels $\phi_x(\alpha_i, \beta_j)$ which generate the sub-matrix $\boldsymbol{\Phi}_x(\mathscr{K}[x_1], \mathscr{K}_x[x_1])$.

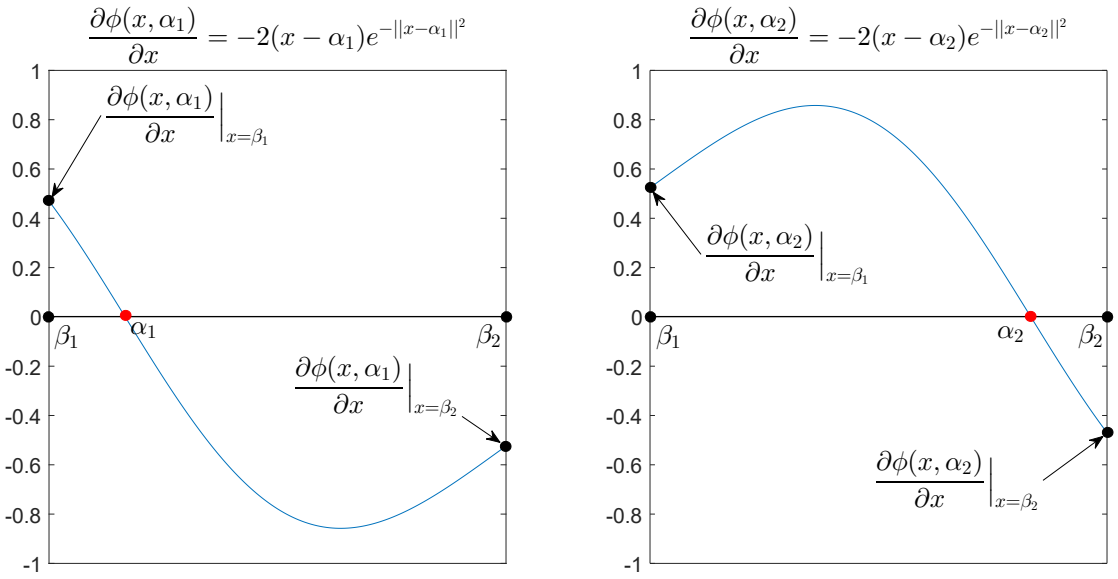


Figure 36: First-order derivatives of the Gaussian functions centered at α_1 (left) and α_2 (right). Also shown are the Gaussian kernels $\phi_x(\beta_i, \alpha_j)$ which generate the sub-matrix $\mathbf{\Phi}_x(\mathscr{K}_x[x_1], \mathscr{K}[x_1])$.

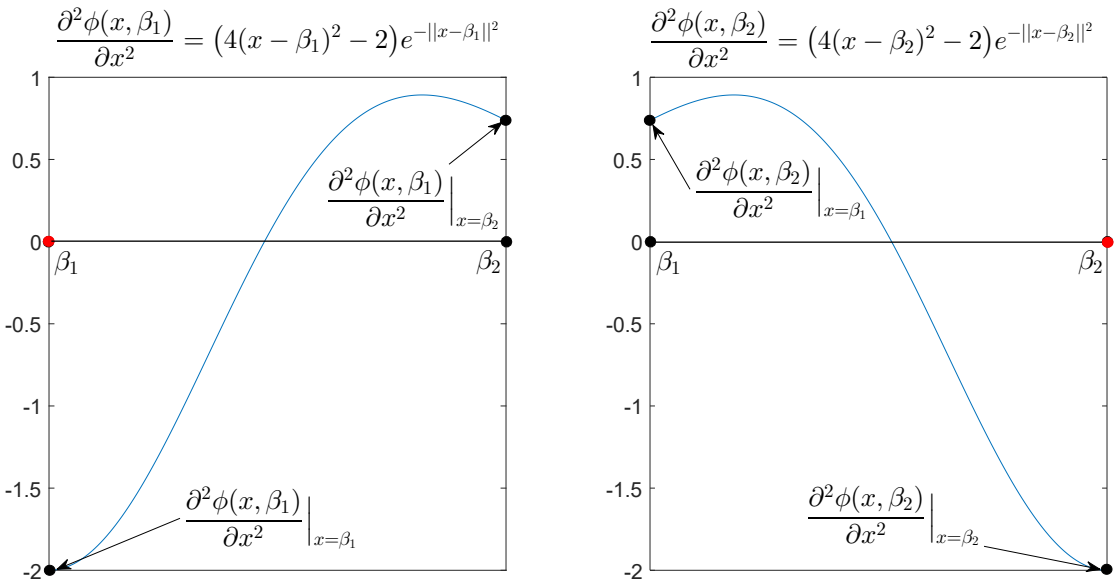


Figure 37: Second-order derivatives of the Gaussian functions centered at β_1 (left) and β_2 (right). Also shown are the Gaussian kernels $\phi_{xx}(\beta_i, \beta_j)$ which generate the sub-matrix $\mathbf{\Phi}_{xx}(\mathscr{K}_x[x_1], \mathscr{K}_x[x_1])$.

Appendix C

Closed-Form Solution of the LATIN Test Problem

General symbols:

Symbol	**Definition and (Units)**
erfc($\square$)	complementary error function of argument $\square$
$l = 10$	length of the one-dimensional domain (m)
$t \in [0, 10]$	time (s)
$u_{SI}(x,t)$: $\mathbb{R}^+$ x $\mathbb{R}^+ \mapsto \mathbb{R}^+$	semi-infinite temperature field in continuous spacetime (K)
$u_0 = 0$	prescribed temperature field at $t = 0$ s (K) **and** prescribed temperature at $x \to \infty$ (K)
$\widetilde{u}(t) : \mathbb{R}^+ \mapsto \mathbb{R}^+$	prescribed Dirichlet temperature in continuous time at $x = 0$ m (K)
$x \in [0, 10]$	position (m)

Greek symbols:

Symbol	**Definition and (Units)**
$\alpha = 1$	thermal diffusivity of the domain (m^2s^{-1})
$\kappa = 1$	thermal conductivity of the domain ($Wm^{-1}K^{-1}$)
$\tau = 10$	quantity of time-steps

Given the simple geometry and boundary/initial conditions of the LATIN test problem (see Figure 2 and Equation (23)), an analytical solution for $u(x,t)$ may be obtained using the separation of variables technique on the heat equation. However, the resulting expression for $u(x,t)$ is not closed-form (i.e. it involves an infinite sum) and is therefore unwieldy to use in practice. A simpler, yet approximate solution may be achieved by constraining the thermal flux at the left end of the domain as $q(x=0,t)$ = -1 Wm^{-2}. Conversely, the right end is allowed to extend to infinity with a temperature of $u(x \to \infty) = u_0$, where u_0 is also the initial domain temperature. In a physical sense, this semi-infinite approach means that the thermal flux at $x =$ 0 m has no effect on the opposite end at $x \to \infty$. Given the parabolic nature of the transient heat equation and the domain of finite length studied in Chapter 4, this stipulation cannot be absolutely satisfied. However, with the following three assumptions the semi-infinite solution can be considered nearly exact:

1. The domain length l is sufficiently large.
2. The simulation duration $t_\tau - t_0$ is sufficiently small.
3. The thermal diffusivity α of the material is sufficiently small.

The semi-infinite temperature distribution which satisfies the assumptions above is expressed, in closed-form, as follows

$$u_{SI}(x,t) = u_0 - \frac{2}{\kappa}\sqrt{\frac{\alpha t}{\pi}}\exp\left(\frac{-x^2}{4\alpha t}\right) + \frac{x}{\kappa}\mathrm{erfc}\left(\frac{x}{2\sqrt{\alpha t}}\right) \tag{115}$$

where κ is the thermal conductivity of the domain. The mathematics literature surveyed in this thesis presents the semi-infinite solution as above, with a prescribed flux at $x = 0$ m and prescribed temperature at $x \to \infty$. In order to match the BCs in Equation (23), a minor transformation is applied to (115) which effectively reverses the domain

$$u_{SI}(x,t) = u_0 - \frac{2}{\kappa}\sqrt{\frac{\alpha t}{\pi}}\exp\left(\frac{-(l-x)^2}{4\alpha t}\right) + \frac{(l-x)}{\kappa}\text{erfc}\left(\frac{l-x}{2\sqrt{\alpha t}}\right) \tag{116}$$

With respect to Chapter 4, the Neumann BC and initial condition are respectively prescribed as -1 Wm^{-2} and $u_0 = 0$ K. The Dirichlet BC calls for a *nominal* value of $\widetilde{u} = u_0 = 0$ K and is trivially simple to enforce in a numerical solution of the problem. However, satisfaction of this same constraint in the semi-infinite solution is dictated by the three conditions on the previous page. Therefore, *in practice* the Dirichlet BC is prescribed as $\widetilde{u}(t) = u_{SI}(x = 0, t)$ in order to force the numerical and semi-infinite temperatures to match at $x = 0$ m for each time-step.

Appendix D

LATIN Sample Calculations

General symbols:

Symbol	Definition and (Units)
$\|\| \cdot \|\|_2$	Euclidean norm
$\mathbf{0}^{10x10}$	matrix of zeros of size 10 x 10
$\mathbf{A} \in \mathbb{R}^{10x10}$	discrete gradient operator (m^{-1})
$c_p = 1$	isotropic specific isobaric heat capacity of the domain ($Jkg^{-1}K^{-1}$)
$\mathbf{C}^{-1} \in \mathbb{R}^{10x10}$	inverse thermal conductivity tensor ($W^{-1}mK$)
$d = 2$	polynomial degree
$\mathbf{F}_1 \in \mathbb{R}^{10x10}$	matrix representing the Dirichlet BCs (Km^{-1})
$\mathbf{F}_2 \in \mathbb{R}^{10x10}$	matrix representing the sensible heat rate, volumetric source/sink magnitudes, and Neumann BCs (J)
grad($\square$)	gradient operator on the field or vector $\square$
$\mathbf{G}$	Gramian matrix

$\mathbf{I}^m$	identity matrix of size m x m
$k = 3$	quantity of data centers used in $\mathscr{K}[\mathbf{r}]$
$\mathscr{L}$	continuous, linear differential operator
$l = 10$	length of the one-dimensional domain (m)
$m = 10$	quantity of free line segments in the domain
$\widetilde{m} = 1$	quantity of surfaces with prescribed Neumann BCs
$M = m + \widetilde{m}$	total quantity of free line segments and prescribed surfaces
$\mathbf{M} \in \mathbb{R}^{10\text{x}10}$	mass matrix (JK^{-1})
$n = 10$	quantity of free nodes in the domain
$\widetilde{n} = 1$	quantity of nodes with prescribed Dirichlet BCs
$N = n + \widetilde{n}$	total quantity of nodes
$p \in \mathbb{N}$	iteration number of the LATIN algorithm
$q(x,t)$: $\mathbb{R}^+$ x $\mathbb{R}^+ \mapsto \mathbb{R}$	flux field in continuous spacetime (Wm^{-2})
$q_{i,k} \in \mathbb{R}$	flux at the i^{th} line segment and k^{th} time-step (Wm^{-2})
$\widetilde{\mathbf{Q}} \in \mathbb{R}^{1\text{x}10}$	prescribed Neumann flux field in discrete spacetime (Wm^{-2})
$\widehat{\mathbf{Q}}^p \in \mathbb{R}^{10\text{x}10}$	KA flux field at the p^{th} LATIN iteration (K)
$\widehat{\mathbf{Q}}^{p+1/2} \in \mathbb{R}^{11\text{x}10}$	KA flux field at the $(p+1/2)^{th}$ LATIN iteration (K)
$\overline{\mathbf{Q}}^{p+1/2} \in \mathbb{R}^{10\text{x}10}$	SA flux field at the $(p+1/2)^{th}$ LATIN iteration (K)

$\mathbf{S} = \mathbf{0}^{10\text{x}10}$	volumetric source/sink field in discrete spacetime (Wm^{-3})
$t \in [0, 10]$	time (s)
$u(x,t)$: $\mathbb{R}^+$ x $\mathbb{R}^+ \mapsto \mathbb{R}^+$	temperature field in continuous spacetime (K)
$\widehat{u}_{i,k} \in \mathbb{R}^+$	KA temperature at the i^{th} node and k^{th} time-step (K)
$u_{SI}(x = 0, t) \in \mathbb{R}^+$	semi-infinite temperature field in continuous time at $x = 0$ m (K)
$(\mathbf{U}^\dagger)^p \in \mathbb{R}^{10\text{x}10}$	temperature lag matrix at the p^{th} LATIN iteration (K)
$\widetilde{\mathbf{U}} \in \mathbb{R}^{1\text{x}10}$	prescribed Dirichlet temperature field in discrete spacetime (K)
$x \in [0, 10]$	position (m)
$x_i^{\pm} \in \mathbb{R}^+$	$x_i^+ = \lim_{\epsilon \to 0} x_i + \epsilon,\ x_i^- = \lim_{\epsilon \to 0} x_i - \epsilon$ (m)

Greek symbols:

Symbol	**Definition and (Units)**
$\Delta t = 1$	uniform time-step size (s)
$\Delta x = 1$	uniform line segment length (m)
$\epsilon = 0.5$	RBF shape factor
$\kappa = 1$	isotropic thermal conductivity of the domain ($\text{Wm}^{-1}\text{K}^{-1}$)
$\Gamma = \partial\Omega$	closed boundary of Ω
$\Gamma_D \subseteq \Gamma$	sub-boundary where the Dirichlet BCs are prescribed

$\Gamma_N \subseteq \Gamma$	sub-boundary where the Neumann BCs are prescribed
$\Omega \subseteq \mathbb{R}^+$	one-dimensional domain
$\widetilde{\phi}_1^u(x) :$ $\mathbb{R}^+ \mapsto \mathbb{R}$	basis function for the space component of $u(x,t)$. Uses a local coordinate system with its origin at the Dirichlet boundary node $x_1 = 0$ m
$\phi_i^u(x) :$ $\mathbb{R}^+ \mapsto \mathbb{R}$	basis function for the space component of $u(x,t)$. Uses a local coordinate system with its origin at x_i in space
$\phi_i^q(x)$ $\mathbb{R}^+ \mapsto \mathbb{R}$	basis function for the space component of $q(x,t)$. Uses a local coordinate system with its origin at x_i in space
$\widetilde{\phi}_{11}^q(x) :$ $\mathbb{R}^+ \mapsto \mathbb{R}$	basis function for the space component of $q(x,t)$. Uses a local coordinate system with its origin at the Neumann boundary surface $x_{11} = 10$ m
$\rho = 1$	isotropic density of the domain (kgm^{-3})
$\tau = 10$	number of time-steps

D.1 Mixed FEM Discretization

The following conditions and properties are summarized from Section 4.1 and are used in the calculations below:

Boundary and Initial Conditions

Dirichlet BC: $u(x = 0, t) = u_{SI}(x = 0, t)$
Neumann BC: $q(x = 10, t) = -1$ Wm^{-2}
Initial: $u(x, t = 0) = 0$ K

Thermal Properties

$\kappa = 1$ Wm^{-1}K^{-1}
$\rho = 1$ kgm^{-3}
$c_p = 1$ Jkg^{-1}K^{-1}

Discretization Properties

$N = 11$ nodes
$\widetilde{n} = 1$ Dirichlet boundary node at $x_1 = 0$ m
$n = 10$ free nodes at $x_2 = 1$ m, ..., $x_{11} = 10$ m
} Dual Grid

$M = 11$ line segments each of length $\Delta x = 1$ m
$m = 10$ free line segments connecting nodes x_1 to $x_2, ..., x_{10}$ to x_{11}
$\widetilde{m} = 1$ Neumann boundary surface at $x_{11} = 10$ m
} Primal Grid

$\Delta t = 1$ s
$\tau = 10$ time-steps

The space-component of the temperature field is spanned by $N = 11$ piecewise linear "hat functions": $\widetilde{n} = 1$ function for the Dirichlet constrained node and $n = 10$ functions for the free nodes

$$\widetilde{\phi}^{u}_{i=1}(x) = \begin{cases} 0, \ x = x_1 \cup x \in (x_2^+, x_{11}] \\ 1 - \dfrac{x - x_1}{\Delta x}, \ x \in (x_1^+, x_2^-] \end{cases}$$

$$\phi^{u}_{i}(x) = \left| \begin{cases} 0, \ x \in [x_1, x_{i-1}^-] \\ \dfrac{x - x_{i-1}}{\Delta x}, \ x \in (x_{i-1}^+, x_i^-] \\ 1 - \dfrac{x - x_i}{\Delta x}, \ x \in (x_i^+, x_{i+1}^-] \\ 0, \ x \in (x_{i+1}^+, x_{11}] \end{cases} \right| i = 2, ..., 10$$

$$\phi^{u}_{i=11}(x) = \begin{cases} 0, \ x \in [x_1, x_{10}^-] \\ \dfrac{x - x_{10}}{\Delta x}, \ x \in (x_{10}^+, x_{11}] \end{cases}$$

Likewise, the space-component of the flux field is spanned by $M = 11$ one-dimensional, piecewise (discontinuous) constant functions: $m = 10$ functions for the free line segments and $\widetilde{m} = 1$ function for the Neumann constrained surface

$$\phi^{q}_{i}(x) = \left| \begin{cases} 0, \ x \in [x_1, x_i^-] \\ 1, \ x \in (x_i^+, x_{i+1}^-] \\ 0, \ x \in (x_{i+1}^+, x_{11}] \end{cases} \right| i = 1, ..., 10$$

$$\widetilde{\phi}^{q}_{11}(x) = \begin{cases} 0, \ x \in [x_1, x_{11}^-) \\ 1, \ x = x_{11} \end{cases}$$

These basis functions are then used to compute three Gramian matrices, with the ultimate objective of computing the $\mathbf{C}^{-1}$, $\mathbf{A}$, and $\mathbf{M}$ matrices on the left-hand side of Equation (24). These Gramian matrices follow the structure of Equation (97)

$$\mathbf{G}(\boldsymbol{\phi}^q, \text{grad}(\boldsymbol{\phi}^u), \Omega) = \frac{1}{\Delta x} \underbrace{\begin{bmatrix} 1 & 0 & 0 & \cdots & 0 & 0 & 0 \\ -1 & 1 & 0 & \cdots & 0 & 0 & 0 \\ 0 & -1 & 1 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & -1 & 1 & 0 \\ 0 & 0 & 0 & \cdots & 0 & -1 & 1 \end{bmatrix}}_{\text{size: } m \text{ x } n \ = \ 10 \text{ x } 10}$$

$$\mathbf{G}(\boldsymbol{\phi}^q, \boldsymbol{\phi}^q, \Omega) = \mathbf{I}^{m=10}$$

$$\mathbf{G}(\boldsymbol{\phi}^u, \boldsymbol{\phi}^u, \Omega) = \mathbf{I}^{n=10}$$

$\mathbf{C}^{-1}$, $\mathbf{A}$, and $\mathbf{M}$ then follow trivially from Equation (19). Evaluation of the $\mathbf{F}_1$ and $\mathbf{F}_2$ matrices on the left-hand side of Equation (24) requires two more Gramian matrices

$$\mathbf{G}(\boldsymbol{\phi}^q, \widetilde{\boldsymbol{\phi}}^u, \Gamma_{\mathfrak{D}}) = \frac{1}{\Delta x} \begin{bmatrix} 1 & 0 & \cdots & 0 & 0 \end{bmatrix}^{\text{T}}, \quad \mathbf{G}(\boldsymbol{\phi}^u, \widetilde{\boldsymbol{\phi}}^q, \Gamma_{\mathfrak{N}}) = \frac{1}{\Delta x} \begin{bmatrix} 0 & 0 & \cdots & 0 & 1 \end{bmatrix}^{\text{T}}$$

which are of respective sizes m x $\widetilde{n} = 10$ x 1 and n x $\widetilde{m} = 10$ x 1. $\mathbf{F}_1$ and $\mathbf{F}_2$ then follow from Equation (20) in conjunction with the prescribed Dirichlet temperature field $\widetilde{\mathbf{U}}$, the prescribed Neumann flux field $\widetilde{\mathbf{Q}}$, and the temperature lag matrix at the p^{th} LATIN iteration $(\mathbf{U}^\dagger)^p$

$$\widetilde{\mathbf{U}} = \underbrace{\begin{bmatrix} u_{SI}(x=0,t_1) & u_{SI}(x=0,t_2) & \cdots & u_{SI}(x=0,t_9) & u_{SI}(x=0,t_{10}) \end{bmatrix}}_{\text{size: } \widetilde{n} \text{ x } \tau \ = \ 1 \text{ x } 10} \text{(K)}$$

$$\widetilde{\mathbf{Q}} = \underbrace{\begin{bmatrix} -1 & -1 & \cdots & -1 & -1 \end{bmatrix}}_{\text{size: } \widetilde{m} \text{ x } \tau \ = \ 1 \text{ x } 10} (\mathrm{Wm}^{-2})$$

$$(\mathbf{U}^{\dagger})^{p} = \underbrace{\begin{bmatrix} 0 & \widehat{u}^{p}_{2,1} & \cdots & \widehat{u}^{p}_{2,8} & \widehat{u}^{p}_{2,9} \\ 0 & \widehat{u}^{p}_{3,1} & \cdots & \widehat{u}^{p}_{3,8} & \widehat{u}^{p}_{3,9} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & \widehat{u}^{p}_{10,1} & \cdots & \widehat{u}^{p}_{10,8} & \widehat{u}^{p}_{10,9} \\ 0 & \widehat{u}^{p}_{11,1} & \cdots & \widehat{u}^{p}_{11,8} & \widehat{u}^{p}_{11,9} \end{bmatrix}}_{\text{size: } n \text{ x } \tau \ = \ 10 \text{ x } 10} (\mathrm{K})$$

Note that the volumetric source/sink magnitude matrix $\mathbf{S}$ has been set to $\mathbf{0}^{10\text{x}10}$ in accordance with the test problem in Chapter 4. The first column of $(\mathbf{U}^{\dagger})^{p}$ is constrained by the initial condition and the other entries are set by the p^{th} guess for the KA temperature field, $\widehat{\mathbf{U}}^{p}$.

On page 50, steps four and five of the LATIN algorithm require two consecutive interpolations of the thermal flux field. The first interpolates the KA flux field $\widehat{\mathbf{Q}}^{p}$ from the $m = 10$ primal line elements to the $N = 11$ dual nodes and is achieved with the following transformation

$$\widehat{\mathbf{Q}}^{p+1/2} = \frac{1}{2} \underbrace{\begin{bmatrix} 3 & -1 & 0 & \cdots & 0 & 0 & 0 \\ 1 & 1 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 1 & 1 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & 1 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 1 & 1 \\ 0 & 0 & 0 & \cdots & 0 & -1 & 3 \end{bmatrix}}_{\text{size: } N \text{ x } m \ = \ 11 \text{ x } 10} \widehat{\mathbf{Q}}^{p}$$

where $\widehat{\mathbf{Q}}^{p+1/2}$ is a matrix of size N x τ representing the KA spacetime flux field at the 11 dual nodes. Observe that the entries of rows one and N of the interpolation matrix have been modified from $1, 1$ to $3, -1$; this is done to *extrapolate* $\widehat{\mathbf{Q}}^{p}$ on to the two boundary nodes at $x_1 = 0$ m and $x_{11} = 10$ m.

The second interpolation projects $\widehat{\mathbf{Q}}^{p+1/2}$ from the $N = 11$ dual nodes back on to the $m = 10$ primal line elements with enforcement of the Neumann boundary condition at $x_{11} = 10$ m. Specifically, this is done by replacing each entry in row N of $\widehat{\mathbf{Q}}^{p+1/2}$ with $\widetilde{q} = -1$ Wm^{-2} and then applying the following transformation

$$\overline{\mathbf{Q}}^{p+1/2} = \frac{1}{2} \underbrace{\begin{bmatrix} 1 & 1 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 1 & 1 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & 1 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 1 & 1 \end{bmatrix}}_{\text{size: } m \text{ x } N \ = \ 10 \text{ x } 11} \widehat{\mathbf{Q}}^{p+1/2}$$

which gives the SA spacetime flux field $\overline{\mathbf{Q}}^{p+1/2}$ of size m x $\tau = 10$ x 10.

D.2 RBF Discretization

In Section 4.2, the following changes are made from the FEM discretization: 1) the $m = 10$ primal line segments are replaced with data centers located at the line segment centroids, symbolized by $\widehat{x}_j$ and 2) the ϕ^u hat functions and ϕ^q piecewise constant functions are replaced with Gaussian RBKs

$$\phi_i^u(x) = \exp[-\epsilon^2(||x - x_i||_2)^2], \quad x_{i=1,...,11} = [0, ..., 10] \text{ m}$$
$$\phi_j^q(x) = \exp[-\epsilon^2(||x - \widehat{x}_j||_2)^2], \quad \widehat{x}_{j=1,...,10} = [0.5, ..., 9.5] \text{ m}$$

The required Gramian matrices are generated by the four steps on page 70: setting $\mathscr{L} \Rightarrow 1$ ultimately gives identical $\mathbf{C}^{-1}$ and $\mathbf{M}$ matrices to those in the FEM formulation, whereas setting $\mathscr{L} \Rightarrow d/dx$ with stencil size $k = 3$, polynomial degree $d = 2$, and shape factor $\epsilon = 0.5$ gives the following for $\mathbf{G}(\boldsymbol{\phi}^q, \text{grad}(\boldsymbol{\phi}^u), \Omega)$

$$\mathbf{G}(\boldsymbol{\phi}^q, \mathrm{grad}(\boldsymbol{\phi}^u), \Omega) =$$

$$\underbrace{\begin{bmatrix} 1.097 & -0.0486 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 \\ -1.097 & 1.049 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 \\ 0.0486 & -1.097 & 1.049 & 0 & \cdots & 0 & 0 & 0 & 0 \\ 0 & 0.0486 & -1.097 & 1.049 & \cdots & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & 0.0486 & -1.097 & 1.049 & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 & 0.0486 & -1.097 & 1.049 \end{bmatrix}}_{\text{size: } m \text{ x } n \ = \ 10 \text{ x } 10}$$

which gives the $\mathbf{A}$ operator via Equation (19). The $\mathbf{G}(\boldsymbol{\phi}^q, \widetilde{\boldsymbol{\phi}}^u, \Gamma_{\mathfrak{D}})$ and $\mathbf{G}(\widetilde{\boldsymbol{\phi}}^u, \boldsymbol{\phi}^q, \Gamma_{\mathfrak{N}})$ matrices are structured like so

$$\mathbf{G}(\boldsymbol{\phi}^q, \widetilde{\boldsymbol{\phi}}^u, \Gamma_{\mathfrak{D}}) = \begin{bmatrix} 1.049 & 0.0486 & 0 & \cdots & 0 \end{bmatrix}^{\mathrm{T}}, \ \mathbf{G}(\widetilde{\boldsymbol{\phi}}^u, \boldsymbol{\phi}^q, \Gamma_{\mathfrak{N}}) = \begin{bmatrix} 0 & \cdots & 0 & 1.049 \end{bmatrix}^{\mathrm{T}}$$

from which the $\mathbf{F}_1$ and $\mathbf{F}_2$ matrices may be computed via Equation (20). The $\widetilde{\mathbf{U}}$, $\widetilde{\mathbf{Q}}$ and $(\mathbf{U}^\dagger)^p$ matrices remain unchanged from Section D.1.

In the RBF formulation, the flux interpolations in steps four and five of the LATIN algorithm are performed as in Subsection 3.1.1. For the same set of RBF parameters specified above, interpolation from the $m = 10$ primal data centers to the $N = 11$ dual data centers is achieved by

$$\widehat{\mathbf{Q}}^{p+1/2} = \underbrace{\begin{bmatrix} 1.644 & -0.788 & 0.144 & 0 & \cdots & 0 & 0 & 0 & 0 \\ 0.394 & 0.712 & -0.106 & 0 & \cdots & 0 & 0 & 0 & 0 \\ -0.106 & 0.712 & 0.394 & 0 & \cdots & 0 & 0 & 0 & 0 \\ 0 & -0.106 & 0.712 & 0.394 & \cdots & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & -0.106 & 0.712 & 0.394 & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 & -0.106 & 0.712 & 0.394 \\ 0 & 0 & 0 & 0 & \cdots & 0 & 0.144 & -0.783 & 1.644 \end{bmatrix}}_{\text{size: } N \text{ x } m \ = \ 11 \text{ x } 10} \widehat{\mathbf{Q}}^{p}$$

The second interpolation projects $\widehat{\mathbf{Q}}^{p+1/2}$ from the $N = 11$ dual data centers back on to the $m = 10$ primal data centers with enforcement of the Neumann boundary condition at $x_{11} = 10$ m. As with the FEM formulation, this is done by replacing each entry in row $N = 11$ of $\widehat{\mathbf{Q}}^{p+1/2}$ with -1 Wm^{-2} and then applying the following transformation

$$\overline{\mathbf{Q}}^{p+1/2} = \underbrace{\begin{bmatrix} 0.394 & 0.712 & -0.106 & 0 & \cdots & 0 & 0 & 0 & 0 \\ -0.106 & 0.712 & 0.394 & 0 & \cdots & 0 & 0 & 0 & 0 \\ 0 & -0.106 & 0.712 & 0.394 & \cdots & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & -0.106 & 0.712 & 0.394 & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 & -0.106 & 0.712 & 0.394 \end{bmatrix}}_{\text{size: } m \text{ x } N \;=\; 10 \text{ x } 11} \widehat{\mathbf{Q}}^{p+1/2}$$

Appendix E

Thermofluid Properties of Water in its Vapour, Liquid, and Saturated States

General symbols:

Symbol	Definition and (Units)
$c_p \in \mathbb{R}^+$	specific isobaric heat capacity of water ($\text{Jkg}^{-1}\text{K}^{-1}$)
$h_{lv} \in \mathbb{R}^+$	heat of vapourization of water (Jkg^{-1})
$p \in \mathbb{N}$	time-step index in the bubble problem
$P_{sat} \in \mathbb{R}^+$	absolute saturation pressure (Pa)
$t^p \in \mathbb{R}^+$	time at the p^{th} time-step of the bubble problem (s)
$u_f(t^p)^\square \in \mathbb{R}^+$	dimensionless saturation temperature at time t^p raised to the power of $\square = 0, ..., 5$
$u_{sat}(t^p) \in \mathbb{R}^+$	saturation temperature at time t^p (K)
$u^\circ_{sat} = 373$	saturation temperature at an ambient pressure of 101.3 kPa (K)

Greek symbols:

Symbol	**Definition and (Units)**
$\kappa \in \mathbb{R}^+$	thermal conductivity of water ($\mathrm{Wm^{-1}K^{-1}}$)
$\mu \in \mathbb{R}^+$	dynamic viscosity of water ($\mathrm{kgm^{-1}s^{-1}}$)
$\rho \in \mathbb{R}^+$	density of water ($\mathrm{kgm^{-3}}$)
$\sigma \in \mathbb{R}^+$	surface tension of liquid water ($\mathrm{Nm^{-1}}$)

Subscripts:

Symbol	**Definition**
l	liquid-phase
v	vapour-phase

The thermofluid properties for water used in Sections 6.1 and 6.2 are given below:

Table 5: Thermofluid properties of water used in the Stefan and interface sucking problems.

κ_l ($\mathrm{Wm^{-1}K^{-1}}$)	0.600
κ_v ($\mathrm{Wm^{-1}K^{-1}}$)	0.0248
ρ_l ($\mathrm{kgm^{-3}}$)	998
ρ_v ($\mathrm{kgm^{-3}}$)	0.0596
$c_{p,l}$ ($\mathrm{Jkg^{-1}K^{-1}}$)	4200
$c_{p,v}$ ($\mathrm{Jkg^{-1}K^{-1}}$)	2034
h_{lv} ($\mathrm{Jkg^{-1}}$)	2251.200E+3

To facilitate comparison with [42] and [43], temperature-dependent thermofluid properties for saturated water [45] are incorporated in Section 6.3. At time-step t^p of the bubble problem, the algorithm on page 130 outputs the saturation temperature inside the bubble, $u_{sat}(t^p)$. This temperature is non-dimensionalized by $u_f(t^p) = u_{sat}(t^p)/u^\circ_{sat}$, where $u^\circ_{sat} = 373$ K is the saturation temperature at an ambient pressure of 101.3 kPa. The thermofluid properties are each expressed as a finite power series in $u_f(t^p)$

$$\begin{bmatrix}\rho_l & \sigma & h_{lv} & \kappa_l & \ln(P_{sat}) & \ln(\rho_v) & \ln(\mu_l)\end{bmatrix}^{\mathrm{T}}(t^p) =$$

$$\begin{bmatrix}
6.028\text{E+3} & -6.460\text{E+3} & 3.270\text{E+3} & -8.108\text{E+2} & 9.807\text{E+1} & -4.668\text{E0} \\
-1.489\text{E-2} & 1.368\text{E-1} & -6.954\text{E-2} & 1.161\text{E-2} & -1.933\text{E-3} & 9.299\text{E-5} \\
1.966\text{E+7} & -2.088\text{E+7} & 1.013\text{E+7} & -2.446\text{E+6} & 2.910\text{E+5} & -1.375\text{E+4} \\
3.218\text{E0} & -4.043\text{E0} & 2.208\text{E0} & -5.533\text{E-1} & 6.593\text{E-2} & -3.071\text{E-3} \\
-5.635\text{E+1} & 5.086\text{E+1} & -1.609\text{E1} & 2.763\text{E0} & -2.474\text{E-1} & 9.091\text{E-3} \\
-6.862\text{E+1} & 5.289\text{E+1} & -1.753\text{E1} & 3.167\text{E0} & -3.015\text{E-1} & 1.198\text{E-2} \\
5.067\text{E+1} & -5.796\text{E+1} & 2.369\text{E1} & -4.981\text{E0} & 5.300\text{E-1} & -2.266\text{E-2}
\end{bmatrix}$$

$$\begin{bmatrix}u_f^0 & u_f^1 & u_f^2 & u_f^3 & u_f^4 & u_f^5\end{bmatrix}^{\mathrm{T}}(t^p)$$

Appendix F

Analytical Solutions for Chapter 6

General symbols:

Symbol	Definition and (Units)
$A, B \in \mathbb{R}^+$	constants
$c_p \in \mathbb{R}^+$	specific isobaric heat capacity of water ($\mathrm{Jkg^{-1}K^{-1}}$)
erf($\square$)	error function of argument $\square$
erfc($\square$)	complementary error function of argument $\square$
$h_{lv} \in \mathbb{R}^+$	heat of vapourization of water ($\mathrm{Jkg^{-1}}$)
$\mathrm{Ja} \in \mathbb{R}^+$	Jakob number
$R(t) \in \mathbb{R}$	instantaneous bubble radius (m)
$\dot{R}(t) \in \mathbb{R}$	instantaneous radial bubble velocity ($\mathrm{ms^{-1}}$)
$s(t) \in \mathbb{R}$	instantaneous interface position relative to the wall (m)
$t \in \mathbb{R}^+$	time (s)

$u(x,t) \in \mathbb{R}^+$	temperature as a function of position (relative to the wall) and time (K)
$u(\xi,t) \in \mathbb{R}^+$	temperature as a function of position (relative to the interface) and time (K)
$v(t) \in \mathbb{R}$	instantaneous velocity ($\mathrm{m}s^{-1}$)
$x \in \mathbb{R}^+$	position measured relative to the wall (m)

Greek symbols:

Symbol	**Definition and (Units)**
$\alpha \in \mathbb{R}^+$	thermal diffusivity of water ($\mathrm{m^2 s^{-1}}$)
$\beta = \dfrac{\rho_v}{\rho_l}$	ratio of vapour and liquid-phase water densities
$\eta = \dfrac{A}{B}$	ratio of constants
$\kappa \in \mathbb{R}^+$	thermal conductivity of water ($\mathrm{Wm^{-1}K^{-1}}$)
$\lambda \in \mathbb{R}^+$	Stefan parameter
$\rho \in \mathbb{R}^+$	density of water ($\mathrm{kgm^{-3}}$)
$\theta = \dfrac{\alpha_v}{\alpha_l}$	ratio of vapour and liquid-phase water thermal diffusivities
$\xi \in \mathbb{R}$	position measured relative to the interface (m)

Subscripts and superscripts:

Symbol	**Definition**
$*$	analytical or semi-analytical value
$+$	dimensionless value
∞	state infinitely far downstream of the interface
l	liquid-phase
s	phase interface
sat	saturation state
v	vapour-phase
w	wall

F.1 Stefan Problem

Analytical solutions [46] for the interface position, interface velocity, and vapour-phase temperature field are presented below

$$s^*(t) = 2\lambda\sqrt{\alpha_v t}$$
$$v_s^*(t) = \lambda\sqrt{\alpha_v} t^{-0.5}$$
$$u_v^*(x,t) = u_w + \frac{u_{sat} - u_w}{\text{erf}(\lambda)} \text{erf}\left(\frac{x}{2\sqrt{\alpha_v t}}\right)$$

where the parameter λ is computed by solution of the following transcendental equation

$$\lambda \text{exp}(\lambda^2)\text{erf}(\lambda) = c_{p,v}\frac{u_w - u_{sat}}{h_{lv}\sqrt{\pi}}$$

F.2 Interface Sucking Problem

The analytical expressions [47] for $s^*(t)$ and $v_s^*(t)$ are identical to those presented in Section F.1. However, the transcendental equation used to solve for the λ parameter is modified as follows

$$\text{exp}(\lambda^2)\text{erf}(\lambda)\left[\lambda - \sqrt{\theta}\frac{\kappa_l}{\kappa_v}\frac{(u_\infty - u_{sat})c_{p,v}\text{exp}(-\lambda^2\beta^2\theta)}{h_{lv}\sqrt{\pi}\text{erfc}(\lambda\beta\sqrt{\theta})}\right] = 0$$

where $\theta = \alpha_v/\alpha_l$ and $\beta = \rho_v/\rho_l$. The analytical liquid-phase temperature field is given below

$$u_l^*(\xi,t) = u_\infty - \left(\frac{u_\infty - u_{sat}}{\text{erfc}(\lambda\beta\sqrt{\theta})}\right)\text{erfc}\left(\frac{\xi}{2\sqrt{\alpha_l t}} + \frac{\lambda(\rho_v - \rho_l)}{\rho_l}\sqrt{\theta}\right)$$

F.3 Bubble Problem

A semi-analytical solution of the bubble problem is presented in [43] and [48] and assumes that the bubble growth is dominated by inertial effects in the early stages and heat diffusion in the later stages. These two assumptions result in the following expression

$$R^+(t^+) = \frac{2}{3}[(t^+ + 1)^{3/2} - (t^+)^{3/2} - 1]$$

where $R^+(t^+)$ and t^+ are the respective dimensionless bubble radius and time as defined by

$$R^+(t^+) = \frac{A}{B^2}R^*(t),\ t^+ = \frac{A^2}{B^2}t$$

where

$$A = \left(\frac{2\rho_v h_{lv}}{3\rho_l}\frac{u_\infty - u_{sat}(t)}{u_{sat}(t)}\right)^{1/2},\ B = \left(\frac{12}{\pi}\mathrm{Ja}^2\alpha_l\right)^{1/2}$$

where Ja is the Jakob number defined as follows

$$\mathrm{Ja} = \frac{\rho_l c_{p,l}}{\rho_v h_{lv}}\big(u_\infty - u_{sat}(t)\big)$$

Combining the four preceding equations and introducing $\eta = A/B$ gives the following expressions for the analytical instantaneous bubble radius and velocity

$$R^*(t) = \frac{2B}{3\eta}\left[\left(\frac{t}{\eta} + 1\right)^{3/2} - \left(\frac{t}{\eta}\right)^{3/2} - 1\right]$$

$$\dot{R}^*(t) = A[(\eta t + 1)^{1/2} - (\eta t)^{1/2}]$$

where $\dot{R}^*(t)$ follows from differentiation of $R^*(t)$ with respect to time.